Model

305890
Spring 2011
5/9/2011
Kyoung Shin Park

Overview

- Model class
  - represents a 3D model composed of multiple ModelMesh objects which may be moved independently.
- ModelMesh class
  - represents a mesh that is part of a Model.
- ModelMeshPart class
  - represents a batch of geometry information to submit to the graphics device during rendering. Each ModelMeshPart is a subdivision of a ModelMesh object.
- To learn how to load the data of an .X file into a Model object and render a 3D model.

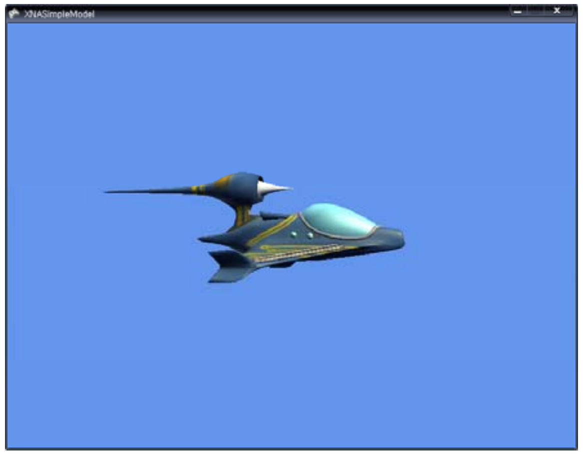
Model Class

- Bones property
  - Gets a collection of ModelBone objects which describe how each mesh in the Meshes collection for this model relates to its parent mesh.
- Meshes property
  - Gets a collection of ModelMesh objects which compose the model. Each ModelMesh in a model may be moved independently and may be composed of multiple materials identified as ModelMeshPart objects.

Model Class

- CopyAbsoluteBoneTransformsTo method
  - Copies a transform of each bone in a model relative to all parent bones of the bone into a given array.
  - When using more complicated models, which often use hierarchical structure (where mesh positions, scales, and rotations are controlled by "bones"), this method ensures that any mesh is first transformed by the bone that controls it, if such a bone exists. The mesh is then transformed relative to the bone transformation.

SimpleModel



Draw a Model

- Load a model using the XNA Framework Content Pipeline
  - You need some art assets (i.e., a 3D model and an associated texture files), and extract its contents to the project Content directory.
  - Add a model "p1_wedge.fbx" in the Content
  - To load the model by using the Content Pipeline
  - Model model = Content.Load<Model>("Models₩₩p1_wedge");
  - This function can load any of the following model formats: FBX and X.

Draw a Model

- Render a model using the XNA Framework Content Pipeline
  - Create a new private method called DrawModel(Model m)

```
private void DrawModel(Model m)  {
      Matrix[] transforms = new Matrix[m.Bones.Count];
      m.CopyAbsoluteBoneTransformsTo(transforms);
      // Draw the model. A model can have multiple meshes, so loop.
      foreach (ModelMesh mesh in m.Meshes)     {
           foreach (BasicEffect effect in mesh.Effects) {
                 effect.EnableDefaultLighting();
                 // ... world, view, projection matrix 중간생략
            }
            // Draw the mesh, using the effects set above.
            mesh.Draw();
      }
 }
```

Draw a Model with a Custom Effect

- Load and render a model using a custom effect without modifying the Content Pipeline.
  - Add a model "Terrain.fbx" in the Content
  - Load the model, typically using the Content Pipeline

```
Model terrain = Content.Load<Model>("Terrain");
Matrix terrainWorld = Matrix.Identity;
Texture2D terrainTex = Content.Load<Texture2D>("TerrainTex");
```

  - Load the effect, typically using the ContentManager

```
Effect effect = Content.Load<Effect>("CustomEffect");
```

Draw a Model with a Custom Effect

- Iterate through each ModelMeshPart in the model, and assign the effect to the Effect property of the ModelMeshPart

```
public static void RemapModel(Model model, Effect effect)
{
    foreach (ModelMesh mesh in model.Meshes) {
        foreach (ModelMeshPart part in mesh.MeshParts) {
            part.Effect = effect;
        }
    }
}
```

- Draw a model

```
foreach (ModelMesh mesh in terrain.Meshes) {
    foreach (Effect effect in mesh.Effects) {
        mesh.Draw();
    }
}
```

Make a Model Move Using Input

- Connect Xbox360 controller or Use keyboard
- Create variables to turn and move the model
- Take input from the user to control the model

Make a Model Move Using Input

- HandleInput

```
private void CheckKeyboardInput(GameTime gameTime) {
    // 중간생략
    if (currentKeyboardState.IsKeyDown(Keys.Left))
        modelRotation += elapsedTime * 3.0f;
    else if (currentKeyboardState.IsKeyDown(Keys.Right))
        modelRotation -= elapsedTime * 3.0f;
    else if (currentKeyboardState.IsKeyDown(Keys.Up)) {
                modelVelocity = Vector3.Forward * 3.0f;
                modelPosition += modelVelocity;
    }
    else if (currentKeyboardState.IsKeyDown(Keys.Down)) {
                modelVelocity = Vector3.Backward * 3.0f;
                modelPosition += modelVelocity;
    }
}
```

Draw Models

- Load and render various models

```
models[0] = Content.Load<Model>("airplane2");
models[1] = Content.Load<Model>("starfish");
models[2] = Content.Load<Model>("cactus");
```