Our estimate of the minimum bandwidth requirment of a flight simulation fly over terrain mdoel for a ideal clien/server rendering algorithm is the same as MPEG1. This can be achieved because the images are highly correlated in time. As a result, MPEG is the ideal compressing for these images.

For sending 30 frame 512×512 RGB uncompressed images per second, the minimum bandwidth is 180Mb per second.

```
        name "Sci-tech.wrl"
        bboxCenter 3153 579 3153
        bboxSize 630 200 630
    }
```

The VRML 2.0 specification provides many new features that can be used to improve the performance of the real-time terrain rendeirng.

ElevationGrid The `ElevationGrid` node provides a convenient way to encode and to present a uniformly sampled 2D surface. The surface is encoded by a matrix of height values in row major order. Since the sampling location is uniformed distributed, the connectivity information is implicitly defined. Hence, VRML 2.0 encoding of a 2D surface is much more compact than VRML 1.0.

NavigationInfo The `NavigationInfo` node contains the description of the physical properties, such as terrain following, speed and visibility, of the viewer and viewing model.

ProximitySensor The `ProximitySensor` node permits the initiation ofactions in the VRML world based on the user position. Hence, we can prefetch the needed terrain data and hide the network latency and transfer time.

Script The `Script` node provides a gateway for the VRML world to incorporate complex animations, which must be defined by a program. The current scripting languages suggested by the specification are Java and Javascript.

Viewpoint The `Viewpoint` node can be used to specify a predefined route for a tour. Using this node, we can limit the flight characteristics of the user's craft, so that the user can experience a smoother presentation on a low-end platform.

VisibilitySensor The `VisibilitySensor` node identifies the visible part of the world. This enables us to order the prefetch process, so that the visible terrain data is fetched with higher priority. In addition, the visibility sensors also permit us to selectively turn on or off the proximity sensor to minimize the processing overhead of the VRML browser.

For the terrain rendering, the accuracy of the terrain model and the texture map is depended on the viewing requirment. For example, distance landscape can be drasticly simplified with out siginifiant degrading the viewing quality. Therefore, a adjustable accuracy of compression facility for the surface and texture will be a useful addition for the VRML 2.0. The accuracy of terrain model is correlate with the number of polygons used by the model. A less accurate model have lees polygon to render, so the rendering frame rate increases. The accuracy of the texture map can be express as the resolution of the map. A lower resolution texture will provide a fussy image. However, it is smaller in size, so that less information is copyed to the texture map buffer. As a result, the rendering speed of a texture mapped polygon increases.

Sicne VRML is downloaded the client computer before rendering, the network bandwidth required for real-time rendeirng a scene is zero.

The terrain data are encode in Digital Elevation Model (DEM) and Landsat images. The DEM forms a polygon based terrain model. The 3D coordinate of the vertices is given by vertices list in the `Coordinate3` node. The polygons are defined by using the index of the vertices list, which is listed in `IndexedFaceSet`.

The Landsat images supplies the color for the terrain model. Using the Landsat images a RGB color can be assigned to each vertex, the color of the vertices is encoded in the `Material` node. The floating point value of the RGB colors is listed as a `diffuseColor` list. The Landsat images can also used directly by VRML as a texture map for the terrain model. The texture can be included as hex value using ASCII, or the texture map can be storted seperatly in a well compressed image file format, such as GIF and JPEG. The basic structure of the terrrain VRML file is,

```
#VRML V1.0 ascii

Separator {
   ShapeHints {
      vertexOrdering COUNTERCLOCKWISE
      faceType CONVEX
      creaseAngle 0.5
   }
   Material {
      diffuseColor [ ... ]
      ambientColor 0.1 0.1 0.1
   }
   Texture2 {
      filename " ... "
# or
      image ...
   }
   TextureCoordinate2 {
      point [ ... ]
   }
   Coordinate3 {
      point [ ... ]
   }
   IndexedFaceSet {
      coordIndex [ ... ]
   }
}
```

Additional objects can be added onto the terrain by using `WWWInline` node. The added object must be stored in a self-contain VRML file. An example of `WWWInline` node structure is shown below

```
 {\tt WWWInline} {
```