PERCEPTION OF DEPTH AND DIMENSION IN THE 3D SIMULATION MODELS

Canan Yemenicioglu[1] and Sinasi Kaya[1]
[1]ITU, Civil Engineering Faculty, Department of Geomatics Engineering 34469 Maslak Istanbul, Turkey
Email: yolcucanan@gmail.com and kayasina@itu.edu.tr

KEY WORDS: Perception of dimension in 3D scenes, perception of depth, perspective projection, graphics engine, visualization, real-time rendering, field of view (FOV), transformation from 3D to 2D, photogrammetry, OpenGL transformation matrices.

ABSTRACT: One of the main rules to realize the realistic perception of the simulation scene is to provide the visualization of the three-dimensional (3D) objects in the same proportions and measurements as it is in the real life. These 3D objects in the scene are converted into 2D images for the projection into the user view with the help of a graphics engine. Nevertheless, the depth perception does not disappear if the transformation is done with the perspective projection methods. That is why the perspective projection is widely used for 3D applications. The concept of the transformation is transferring the position of all vertices from X, Y, Z object coordinates into X, Y and depth buffer (Z) screen coordinates. In fact, these are the transformations which we use in photogrammetry. All X, Y, Z coordinates of the model are scaled into the X, Y, Z screen coordinates, which lie in the -1, +1 interval. In this study, the transformation of different models with the perspective projection method is calculated. The objects are observed from different distances. The dimensions of the models after the calculation with perspective projection and the measurements from the display are compared. In addition to that, it is also observed if different field of view (FOV) values affects the proportions of the model dimensions. As a result, it is seen that the perspective projection is a successful transformation method to represent the 3D models in 2D displays, and different FOV values have no influence on the dimension proportions of the models. Software tools, 3dsMax and Presagis Creator for modeling and editing and Vega Prime for visualization, are used in this work. OpenGL library is used as a basis for the mathematical calculations and transformations.

1. INTRODUCTION

To understand how the object models are presented to the simulation user as they are in the real world, the transformation from 3D space to 2D screen area should be investigated. These transformations are essentially an application of photogrammetry.

There are two main methods to visualize 3D objects on the screen: Orthographic projection and perspective projection (House, 2015; Sun et al., 2015). Orthographic projection results in an unrealistic representation of the object model, therefore it is not preferred for simulation applications. Perspective projection is the main method, which is used in the simulation. With the help of the perspective projection, the object models are scaled down in the simulation scene, as they gain distance from the point of view. This method creates the perception of reality for the projection of the simulation models. Popular graphics/modeling software like Vega Prime or Open Scene Graph use Cartesian coordinate system (Elmqvist and Tsigas, 2007; House, 2015; Presagis, 2016). The graphics engine acts on the assumption that there is a camera on the point of view during the projection of 3D models into the user's sight. All vertices on the 3D space are converted to 2D screen points on runtime during the presentation (David et al., 2004).

2. METHODOLOGY AND APPLICATION

The user's perception takes place on the screen, so it is important to define the mechanism of projection to the screen. The horizontal axis of the screen is defined as X-axis and the vertical axis as Y-axis. The middle of the screen is the point (X=0, Y=0), the bottom left corner is (-1, -1) and upper right corner is (+1, +1). It is also assumed that the camera (point of view) lies on the (0,0,0) and the objects are placed on the Z axis for the depth. The nearest point of the Z is defined as -1 and the farthest point is defined as +1 (This means the direction of +Z is getting far and -Z getting near to the camera).

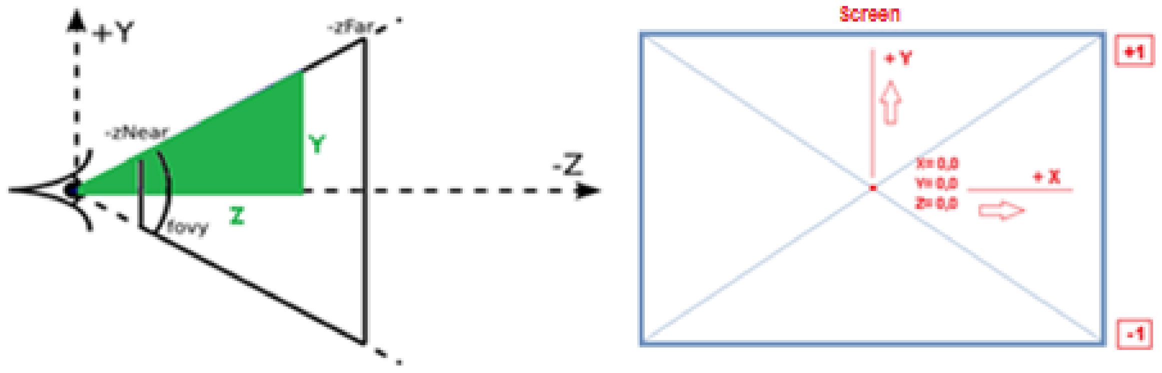


Figure 1. The side view of the camera frustum (right) and screen coordinates (left) (OpenSceneGraph, 2016)

All the vertices, which does not stay inside the defined cube volume with the coordinates from (-1,-1,-1) to (+1,+1,+1), are clipped away from the scene, meaning not displayed. The depth values zNear and zFar define the limitations for the display of the simulation scene. The area between these values is displayed, the outside is clipped.

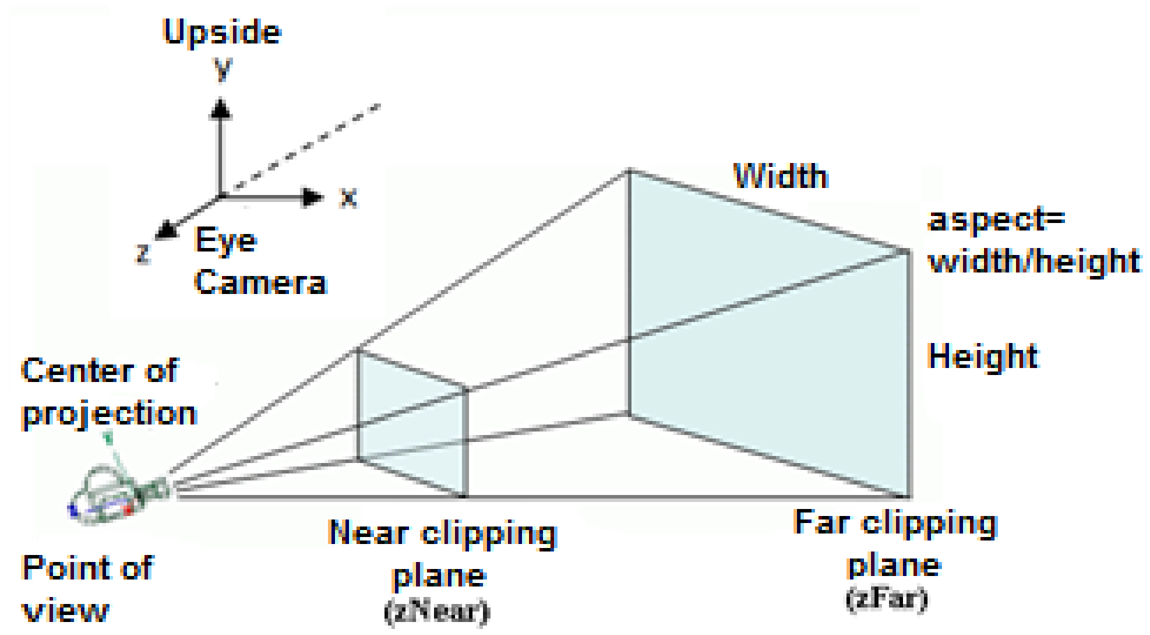


Figure 2. Camera frustum and view area

The vertices are transferred from 3D to 2D with transformation matrices in perspective projection. If the mathematics of the transformation is examined, four parameters are relevant for the further calculations (OpenSceneGraph, 2016).

- **Fovy (Field of view):** Vertical projection angle on the Y axis
- **Aspect ratio:** The ratio of vertical projection angle to the horizontal angle. Can be defined as the variable $Fov_x = Fov_y \ x \ aspect$. This variable has the value of horizontal pixel resolution divided by vertical pixel resolution for a full-screen projection; i.e. for a screen with resolution 640x480, the aspect ratio should be 4/3.
- **zNear:** The inverted Z coordinate of the near clipping plane
- **zFar:** The inverted Z coordinate of the far clipping plane

The calculation of the transformation equations results in the following equations for the screen coordinates:

$$f = 1/\tan\left(\frac{fovy}{2}\right)$$

$$x_{screen} = \frac{fx}{aspect}/-z$$

$$y_{screen} = fy/-z$$

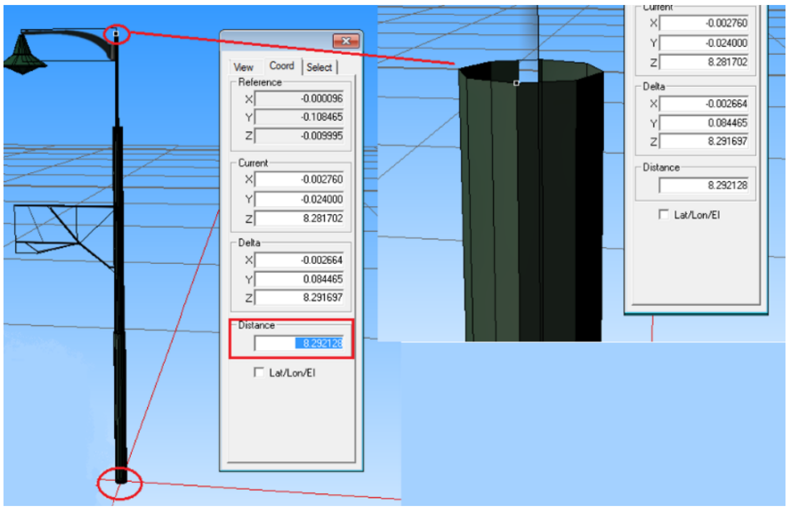
$$z_{screen} = \frac{2.zFar.zNear}{zFar - zNear}/-z - \frac{zFar + zNear}{zNear - zFar}$$

3. RESULTS

The simulation of ground vehicles is a good example to investigate the effect of perspective perception, as the user is observing the scene from a near distance. For this reason, the model for the application of perspective projection is selected from a ground vehicle simulation scene. The aim of this paper is to investigate the accuracy of the model projections, so the necessary steps are explained in this section.

Figure 3. The simulation scene from the user's point of view (Guica, 2016)

The first step is to measure the dimension of the model, which will be transformed to screen coordinates, in the Creator software tool (Katron, 2004). The length of the model (a pole in this example) is measured as **8.29 m** in the simulation scene as seen in Figure 4. The camera position is assumed on the coordinates (0,0,0), and transformation equations are calculated for the perspective projection. The screen resolution for the projection has been taken as 800x600 and vertical projection angle on the Y axis (Fovy) as $\mathbf{60^o}$. The maximum distance between the model and camera (point of view) is 25m, so the clipping values for depth are selected as zNear=1 and zFar =30.



View | Coord | Select

Reference
X -0.000096
Y -0.108465
Z -0.009995

Current
X -0.002760
Y -0.024000
Z 8.281702

Delta
X -0.002664
Y 0.084465
Z 8.291697

Distance
8.292128

Lat/Lon/El

Figure 4. 3D dimensions of the investigated pole model

The next step is the measurement of the model dimension on the screen with distances of 25m and 15m.

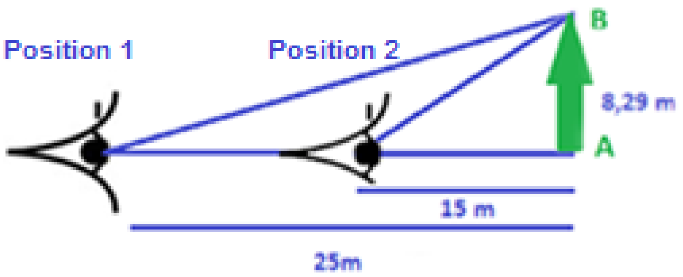


Figure 5. Observation positions of the model

The coordinates of the bottom of the pole (A) and top of the pole (B) for both of the observation points are given as the following on the simulation scene:

$$\mathbf{A_1} = (0\,,0,-25) \quad \mathbf{B_1} = (0,8.29,-25)$$
$$\mathbf{A_2} = (0,0,-15) \quad \mathbf{B_2} = (0,8.29,-15)$$

The last step is calculating the expected dimensions on the screen coordinates for the observation positions (25m and 15m). The lengths are gathered from the distance of top and bottom points of the pole. The ratio of the measurements from the screen (4.10 cm from 25m distance and 6.75 cm from 15m distance) has been compared with the calculations from the transformation equations, and the following error ratio has been gathered:

$$\mathbf{1.Result}: \left(\frac{\textbf{Pole length calculation (25 m)}}{\textbf{Pole length calculation (15 m)}}\right) = \frac{\mathbf{Yv_{B1} - Yv_{A1}}}{\mathbf{Yv_{B2} - Yv_{A2}}} = \frac{\mathbf{0.5743 - 0}}{\mathbf{0.9572 - 0}} = \mathbf{0.6}$$

$$\mathbf{2.Result}: \left(\frac{\textbf{Pole length measurement (25 m)}}{\textbf{Pole length measurement (15 m)}}\right) = \frac{\mathbf{4.10}}{\mathbf{6.75}} = \mathbf{0,6074}$$

Error ratio: (1.Result-2.Result) /1.Result= %1.2345

Different field of view (Fovy) angles has been used to investigate if this variable has an influence on the ratio. The calculations are done with the values $\mathbf{60^o}$, $\mathbf{90^o}$, and $\mathbf{100^o}$. The ratio of the pole lengths does not change with the Fovy angle, as seen in Figure 6.

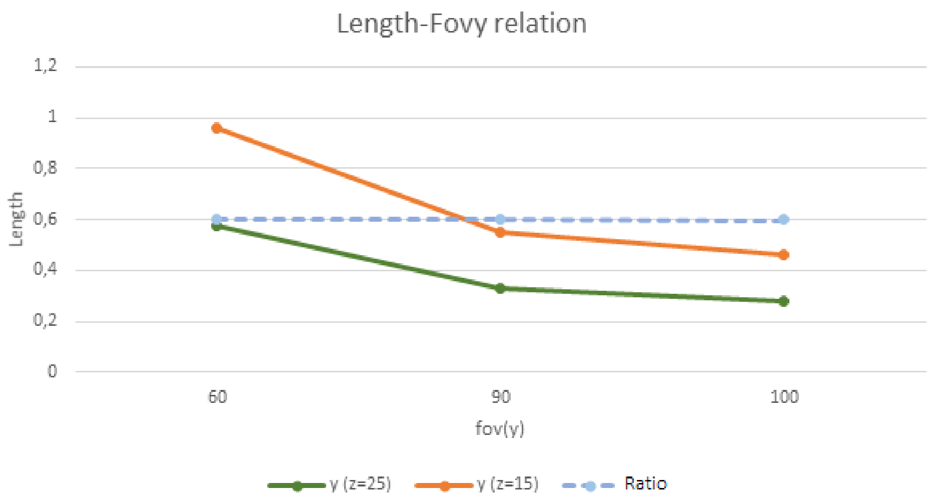


Figure 6. Length/Fovy relation with different angles

Another result of the examinations is that the depth perception of the scene lessens if the model has more distance from the point of view. The graphics software has a better capacity to display the depth differences for the near objects in relation to the far objects. As a result, the object models, which are staying near the far clipping plane, can be sensed by the user as if they are at the same distance, meaning the distances cannot be detected.

4. CONCLUSIONS

The results of this study have shown that the graphics software executes the perspective projection method for the visualization of the models successfully. The error amount, which lies in %1 range, results from the manual measurements on the screen so that it can be ignored. The possible error source on the simulation scene is mostly the 3D model itself. This information is relevant because users of the simulation sense the object model in a relationship with other object models staying nearby. A defected/inaccurate modeled object also affects the sensation of the nearby models, which influence the visual perception of the whole scene. Another result is that the different field of view angles (Fovy) has no effect on the model dimensions. The last point to mention is that the objects staying on the near of the point of view can be perceived better than the objects far away, as the calculation methods are not capable of displaying the differences accurately for the objects near the far clipping plane.

References

David, P., DeMenthon, D., Duraiswami, R., Samet, H., 2004. SoftPOSIT: Simultaneous Pose and Correspondence Determination. Int J Comput Vision, 59(3): pp. 259–284. doi: 10.1023/b:visi.0000025800.10423.1f

Elmqvist, N., Tsigas, P., 2007. View-projection Animation for 3D Occlusion Management. Computers & Graphics, 31, pp. 864-876.

Guica, M., "Calculating the glu Perspective matrix and other OpenGL matrix maths," [Online]. Available: https://unspecified.wordpress.com/2012/06/21/calculating-the-gluperspective-matrix-and-other-opengl-matrix-maths/. [Accessed 14 April 2016].

House, D.H., 2014. Orthographic and Perspective Projection, [Online]. Available: https://people.cs.clemson.edu/~dhouse/courses/405/notes/projections.pdf. [Accessed 10 May 2016].

Katron Inc., Company Presentation, Istanbul, 2004.

OpenSceneGraph Community, "The OpenSceneGraph Project Website," [Online]. Available: http://www.openscenegraph.org/index.php/about/features. [Accessed 5 April 2016].

Presagis, 2016. Overview, http://www.presagis.com/products_services/products/modeling-simulation/visualization/vega_prime#overview. (Accessed 5 April 2016).

Sun, P., Sun, C., Li, W., Wang, P., 2015. A New Pose Estimation Algorithm Using a Perspective-Ray-Based Scaled Orthographic Projection with Iteration. PLoS ONE 10(7): e0134029. doi:10.1371/journal.pone.0134029.