Voxel-Based Game Engine in QBASIC

Purpose

To create a CPU-based 3D voxel renderer for use in potential computer games.

Considerations

Engine should be able to run at 50 fps.
Engine should be customizable from outside of the source code

Specification

Render 3D scenes using single-colored volumetric pixels (voxels)
Small objects (<32*32*32 voxels) should be rendered in real time (~50 fps)
Large backgrounds can be rendered once, and then saved as a 2D image which can be displayed every frame
This means that the background should only be updated periodically
Objects should be stored in files separate from the source code which can be edited by other programs (.csv format used)
All object files should be loaded into RAM before gameplay because this is an expensive process which cannot be completed with any regularity (without sacrificing performance)

Solution:

In order to convert 3D coordinates into a 2D space, the following transformation is used (where (0,0) is the center of the screen):

```
new_x=old_x/old_z
new_y=old_y/old_z
```

A Z-Buffer was used in order to implement occlusion
Every time a voxel is rendered, its distance from the camera is saved to a 2D array called a Z-Buffer
A voxel is only rendered if the value in the Z-Buffer at that screen position is not lower than the value of
the voxel to be rendered.
In the file format of the 3d objects, Each row of the .csv file contains one voxel
There is a column for X, Y, Z, and color.
There is a small editor program which makes it easier to edit these files.