# Algorithmic effects description

## Level of detail

The level of detail effect is visible while moving towards the house. To implement this, a own LevelOfDetailSGNode has been created, which takes three objects, one for each level.

In the render() function it decides which object to render by comparing the current distance from the object to the camera with the level-thresholds.

Here is the rendering decision in pseudo-code:

if (distance > threshold1) { //render level 0 }

else if (distance > threshold2) { //render level 1}

else { //render level 2 }

The level thresholds are constants: threshold1 = 20, threshold2 = 10.

The distance from the object to the camera is calculated as the euclidian distance between two 3D-vectors:

The three levels of the house are as follows:

Level 0: The house consists only of a quad and a roof with a constant material color (brick-red).

Level 1: A texture is mapped onto each wall and on the roof

Level 2: A texture is additionally mapped onto the floor and the ceiling. The front wall gets split into pieces and windows and a door are added. A basement with wall texture is added to compensate the roundness of the planet, so there is no gap between the house and the planet anymore. A swinging lamp is added to the ceiling. A smoking dalek is added inside the house.

## Particle System

The particle system can be seen on the Dalek in the house. The class Particle stores the current position, the start position, the direction, the speed, the start time and a calculated age.

In a method the current position is updated. The position is calculated by adding a portion of age times speed of the direction to the start position.

The age is the time passed between start time and current time.

If the particle reaches a certain age the particle is reset. The start time is set to current time and the current position is recalculated.

In a function depending on the current time the particles are created and updated. New particles are generated until a limit is reached. There are two arrays: one holds the particle objects and one holds the scene graph nodes. A loop updates periodical the position of the particle objects and then translates the particle scene graph nodes to the new position.