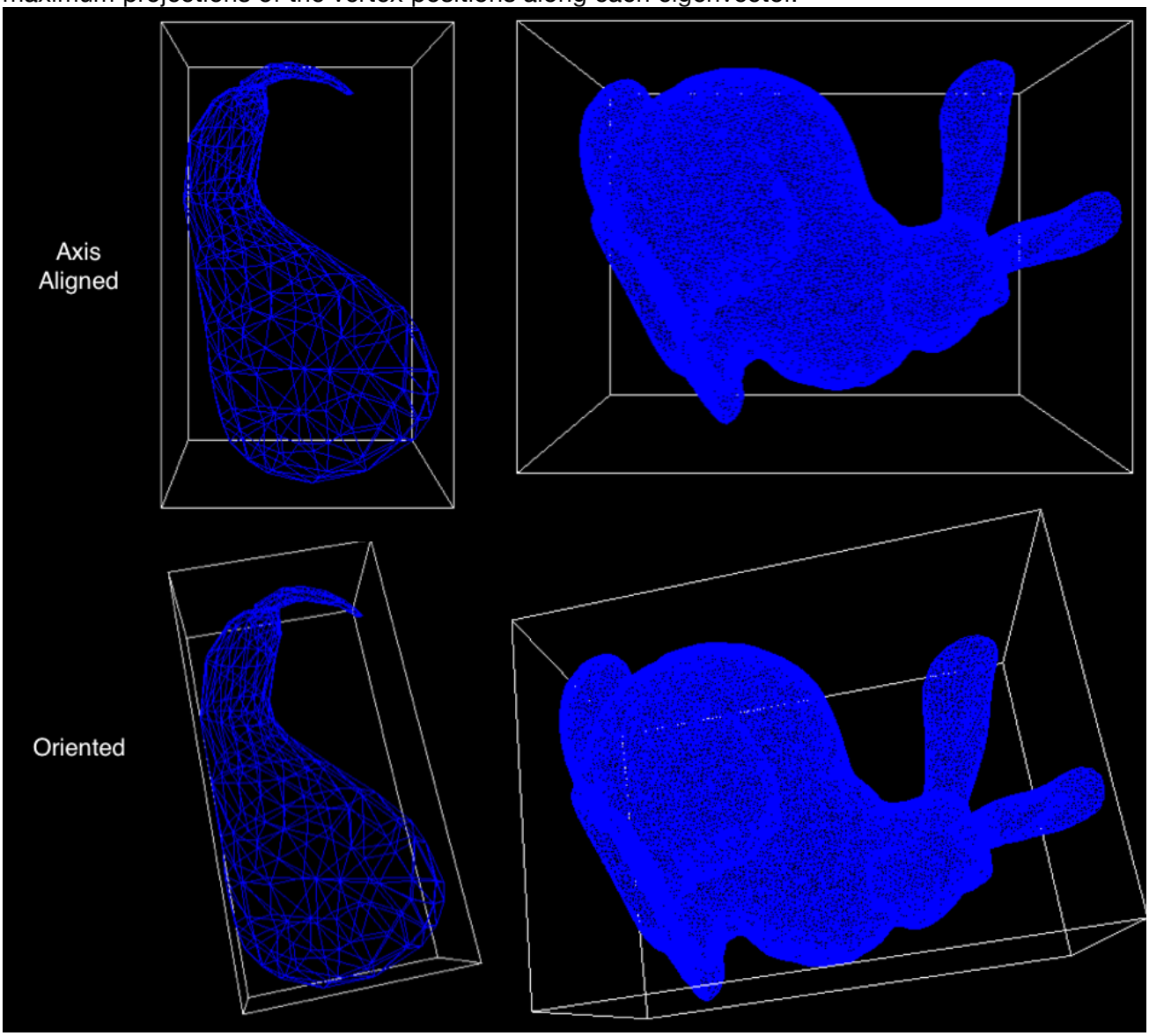
Computing Oriented Bounding Boxes Using Principal Component Analysis

In acceleration data structures such as the Bounding Volume Hierarchy, the design of the bounding boxes themselves play a integral role in quickly determining intersection between objects. An axis aligned bounding box, while simple to build and easy to test for intersection with, does not necessarily fit the objects in it tightly and thus increases the chances of false positive intersections. Oriented bounding boxes reduce the possibility of such false positives by fitting objects they store more tightly. This is achieved by aligning the boxes along the principal axes of the objects.

The principal axes of an object or group of objects can be found by using Principal Component Analysis. It works as follows:

1. Compute the center of mass (mean) of the vertex positions (point cloud).
2. Build the 3x3 covariance matrix of the point cloud data. In code, this translates to evaluating the expression $\sum_i v_i^T v_i \ / \ |V|$, where $v_i = [x_i \ y_i \ z_i]$ is a column vector of the i^{th} vertex position and $|V|$ is the number of vertices. The matrix is real and symmetric.
3. Compute the eigenvectors of the covariance matrix. These serve as the principal axes of the objects.

The extents of the bounding box along each axis can be found by determining the minimum and maximum projections of the vertex positions along each eigenvector.



A point to note. The Principal Component Analysis essentially finds the axes of an ellipsoid that approximates the point cloud data. This works well in most cases, but can result in boxes with larger than expected volumes for symmetric objects like a cube. This is because the approximating ellipsoid for a cube is a sphere, and a sphere does not have a well defined axis.

Implementation: https://github.com/rohan-sawhney/oriented-bounding-box