What was the bug?

The algorithm of constructing points on the base and top circles to generate the light volume through Delaunay algorithm.

```
if strcmp(shape, 'cylinder')
    dir = varargin{2} - varargin{1};
    tangent = dir/norm(dir);
    if(tangent(2) ~= tangent(1))
        normal = [tangent(2) tangent(1) tangent(3)];
    else
        normal = [tangent(1) tangent(3) tangent(2)];
    end
    normal = normal/norm(normal);
    binormal = cross(tangent, normal);
    binormal = binormal / norm(binormal);
    steps=10;
    for i=1:steps
        points(i,:) = [radius*cos(2*pi/steps * i) radius*sin(2*pi/steps * i)];
    end


transformed_points = [];

for  i=1:size(points,1)
        newpoint = origin+points(i,1)*normal + points(i,2)*binormal;
        transformed_points = [transformed_points; newpoint];
end
for  i=1:size(points,1)
        newpoint = origin+points(i,1)*normal + points(i,2)*binormal + norm(dir)*tangent;
        transformed_points = [transformed_points; newpoint];
end

DT = delaunayTriangulation(transformed_points);
```

The original code is using the wrong vectors (normal and binormal) to construct points of the circle.

Solution:

The original code is using two vectors (normal and binormal) to construct points of the circle. The vectors are not the best vectors to work in all kind of geometries.

```
if strcmp(shape, 'cylinder')
    dir = varargin{2} - varargin{1};
    tangent = dir/norm(dir);
    if(tangent(2) ~= tangent(1))
        normal = [tangent(2) tangent(1) tangent(3)];
    else
        normal = [tangent(1) tangent(3) tangent(2)];
    end
    normal = normal/norm(normal);
    binormal = cross(tangent, normal);
    binormal = binormal / norm(binormal);
    trinormal = cross(tangent, binormal);
    trinormal = trinormal / norm(trinormal);
    steps=10;
    for i=1:steps
        points(i,:) = [radius*cos(2*pi/steps * i) radius*sin(2*pi/steps * i)];
    end


 transformed_points = [];

 for  i=1:size(points,1)
        %newpoint = origin+points(i,1)*normal + points(i,2)*binormal;
        newpoint = origin+radius*cos(2*pi/steps * i)*binormal + points(i,2)*trinormal;
        transformed_points = [transformed_points; newpoint];
 end
 for  i=1:size(points,1)
        %newpoint = origin+points(i,1)*binormal + points(i,2)*binormal + norm(dir)*tangent;
        newpoint = waypoint+radius*cos(2*pi/steps * i)*binormal + points(i,2)*trinormal;
        transformed_points = [transformed_points; newpoint];
 end

 DT = delaunayTriangulation(transformed_points);
```

The modified code is using the two correct vectors (binormal and trinormal) to construct points of the circle. Using theses vectors gives the flexibility for the light boundary to be in any configuration.

Underlying Geomatical rule:

Two construct points on a circle having its centre ($\vec{C}$), its radius (r), and the normal vector ($\overrightarrow{tangent}$!), we need to follow these steps:

1-finding to orthonormal vectors that lie in the plane of the circle, perpendicular to each other and to the normal vector.

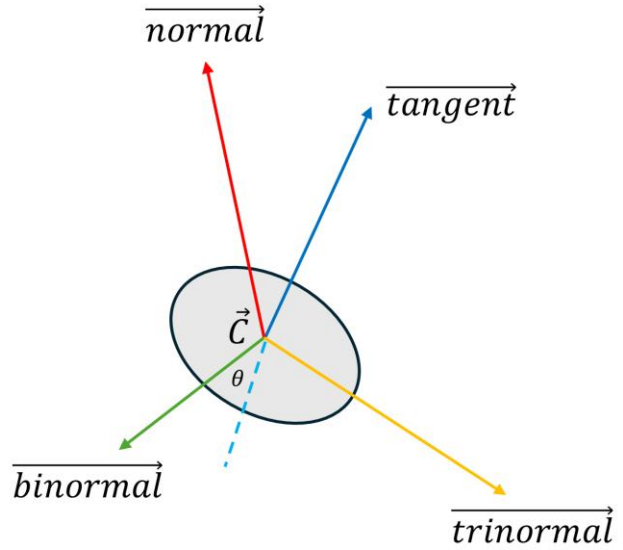
1-1-Choose any vector that is not parallel to $\overrightarrow{tangent}$ and call it $\overrightarrow{normal}$. This vector $\overrightarrow{normal}$ can be arbitrary.

1-2- Find $\overrightarrow{binormal} = \frac{\overrightarrow{tangent} \times \overrightarrow{normal}}{|\overrightarrow{tangent} \times \overrightarrow{normal}|}$. This gives you a unit vector perpendicular to $\overrightarrow{tangent}$.

1-3-Compute $\overrightarrow{trinormal} = \frac{\overrightarrow{tangent} \times \overrightarrow{binormal}}{|\overrightarrow{tangent} \times \overrightarrow{binormal}|}$, which gives you another unit vector in the plane, perpendicular to both $\overrightarrow{tangent}$ and $\overrightarrow{binormal}$.

2-find coordinates of points using the following equation:

$$\overrightarrow{r(\theta)} = \vec{C} + r\left(\cos(\theta)\overrightarrow{binormal} + \sin(\theta)\overrightarrow{trinormal}\right)$$



Acknowledgment:

- ✓ This work was funded by UK Research and Innovation (UKRI) under the UK government's Horizon Europe funding guarantee [grant number 10081458].
- ✓ This work was part of the HORIZON-EIC-2022-TRANSITIONOPEN project PHIRE, Project number: 101113193.