Point Cloud to Sectional Area Curve

Stamatis Stamatelopoulos

June 18, 2023

Contents

Chapter 1

Testing

1.1 `test-1.cpp`

The so-called *extended Wigley Hull form* (Journée 1992) has a simple analytic formulation which permits the exact calculation of it's cross-sectional area curve. We use this example to test the accuracy of `SectionalAreaXwiseYsymmetrical()` to verify. While the standard Wigley hull is a rather poor geometry, in the extended version, by varying the parameters c_1, c_2 and c_3 outside their prescribed domains $[0, 1]$ richer geometries can be generated even if they no longer represent a hull-form. The one used here is retrieved from the sample space by setting $L = 2$, $B = 0.5$, $T = 0.2$, $c_1 = 5.2$, $c_2 = 2.1$ and $c_3 = 2.3$ (see Figure 1.1).

Figure 1.1: Geometry used in `test-1.cpp`. $L = 2$, $B = 0.5$, $T = 0.2$, $c_1 = 5.2$, $c_2 = 2.1$ and $c_3 = 2.3$

The `SectionalAreaXwiseYsymmetrical()` procedure works with a point cloud as input, in order to calculate the cross sectional areas, *along the X-direction*, of the underlying geometry. Naturally, said geometry must comply with certain assumptions: given a solid bounded by a surface,

1. It must be symmetrical with respect to the XZ plane passing through the origin and the point cloud must all lie on one side of it (either positive Y or negative Y)

2. It does not have to be convex, however it must be star-convex with respect to the XZ plane: for every point $p = (p_1, p_2, p_3)$ on the bounding surface, the line between $(p_1, 0, p_3)$ and p must be completely contained in the solid

3. The point cloud provided must be entirely on the boundary surface (i.e. not on the solid whose boundary is the given surface)

The point cloud provided will directly affect the accuracy of the result, in the sense that uniform point clouds might perform better than arbitrarily generated ones. Nevertheless, for this experiment, the point clouds generated are randomly sampled from the boundary surface's domain.

Now, the scheme employed in `SectionalAreaXwiseYsymmetrical()` has the following seven steps

1. Identify the domain of the sectional area curve, parametrized in accordance with the actual surface coordinates

2. Discretize the identified domain in N sub-domains $[a_i, b_i]$ $i = 1, ..., N$, each referring to a specific section. Specifically, given the domain $[a, b]$, $N \in \mathbb{N}$ and $\lambda \in \mathbb{R}$, first generate $t_i = a + (b - a)(i - 1)/(N - 1)$ for $i = 1, ..., N$. Then, set $a_0 = a$, $b_N = b$, $l = (t_1 - t_0)\lambda$ and $a_i = t_{i-1} - \lambda$, $b_i = t_i + \lambda$ for all other i.

3. Group the point cloud in terms of this discretization, assigning every point $p = (p_1, p_2, p_3)$, to the sub-domain $[a_i, b_i]$ where $a_i \leqslant p_2 < b_i$, thereby generating N such groupings

4. For every grouping, project all points on the plane $(a_i/2 + b_i/2, u, v)$ along $(1, 0, 0)$

5. For every grouping, sort points Z-wise

6. For every grouping sum up the rectangles formed by consecutive points p, q by the following four vertices: (pq), $(p\tilde{p})$, $(q\tilde{q})$ and $(\tilde{p}\tilde{q})$, where for $x = (x_1, x_2, x_3)$, $\tilde{x} = (x_1, 0, x_3)$

7. For every grouping multiply the total resulting area by 2

We proceed by a brief investigation of the effect of the following parameters, on the accuracy of the approximation: (1) size of the generated point cloud (2) number of points to evaluate the sectional area at (3) $\lambda \in [0, 1]$ to determine for each sectional area, how many neighboring points are considered. From the relevant experimentation, it seems that the determination of this parameters is an information issue: the more information (large point cloud), the more sectional-area curve points can be requested with sufficient accuracy, which is to be expected. However, this means that depending on the application (i.e. calculation of sectional area curve derivatives via finite differences), a relatively small point cloud might be sufficient. We begin with the generation of a point cloud in 3000 points, each randomly sampled from the surface's domain (see Figures 1.2 and 1.3).

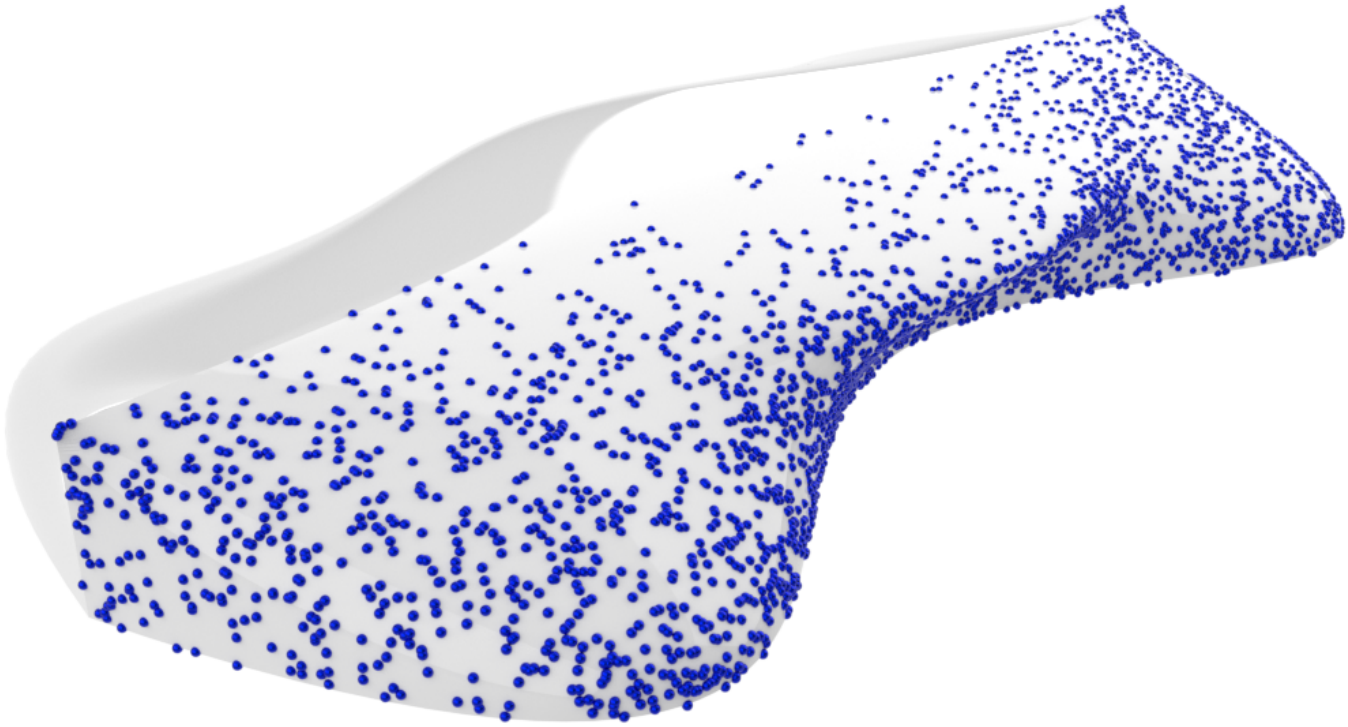

Figure 1.2: 3000 points randomly generated on the surface, top view

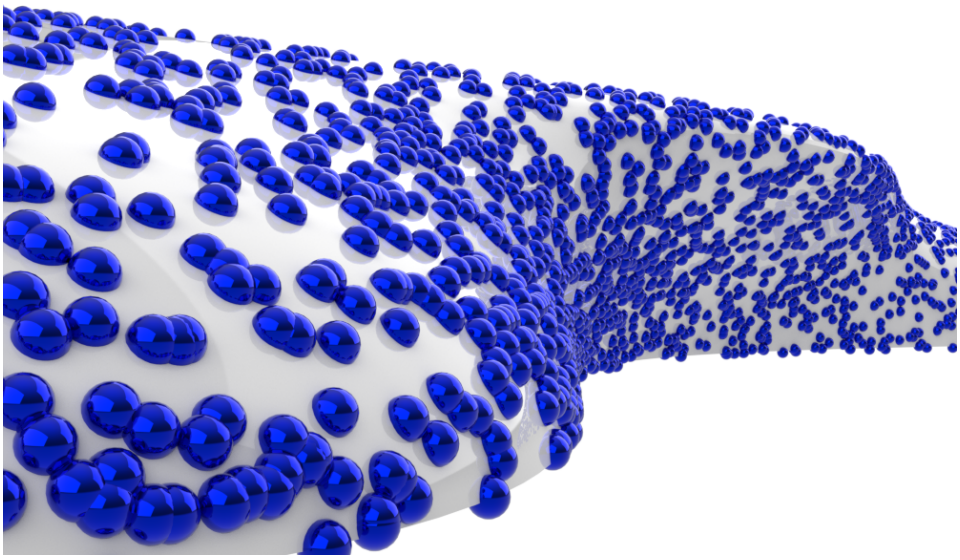

Figure 1.3: 3000 points randomly generated on the surface, bottom view

Then, let $N = 10$ and $\lambda = 1.0$, so that all the points in the point cloud are used. Looking in Figure 1.4, this selection of N, λ results in an L^∞ error of 0.00711 at the calculated points. Part of this error is caused by using points far away from the longitudinal point of evaluation to evaluate the sectional area curve. If we instead limit the this selection by setting $\lambda = 0.78$, the error can be reduced to 0.00332 (see Figure 1.5).

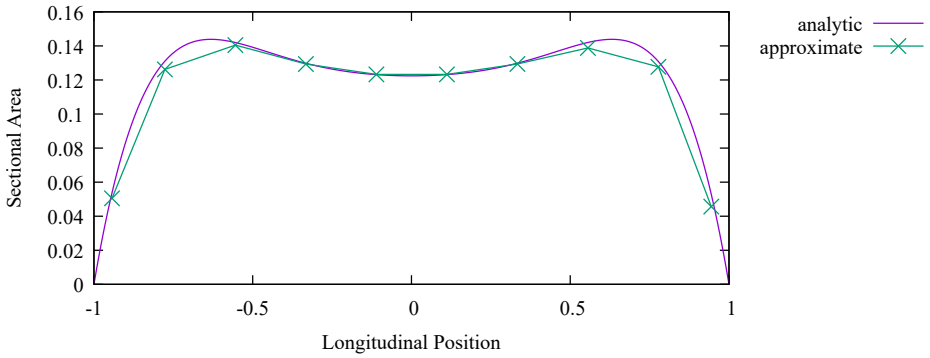

Figure 1.4: Sectional area curve at $N = 10$ points, of the geometry depicted in Figure 1.1, using 3000 randomly sampled points, and $\lambda = 1.0$. The L^∞ error at the evaluated points is 0.00711

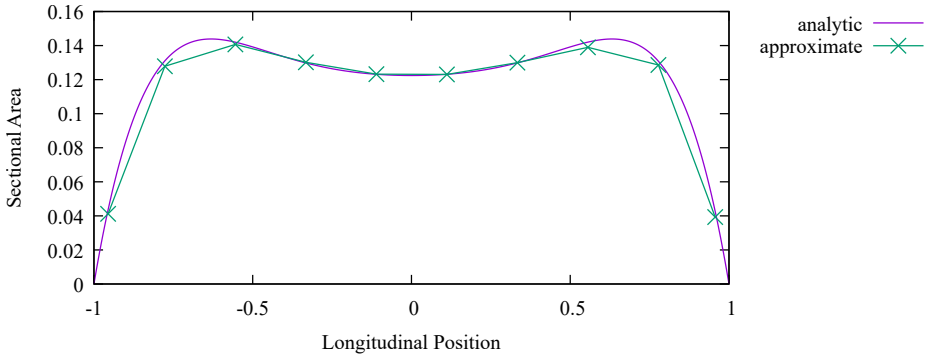

Figure 1.5: Sectional area curve at $N = 10$ points, of the geometry depicted in Figure 1.1, using 3000 randomly sampled points, and $\lambda = 0.78$. The L^∞ error at the evaluated points is 0.00332

Finally, Figures 1.6 and 1.7 illustrate this selection, where the bronze cubes are the points that were used to calculate each cross section's area.

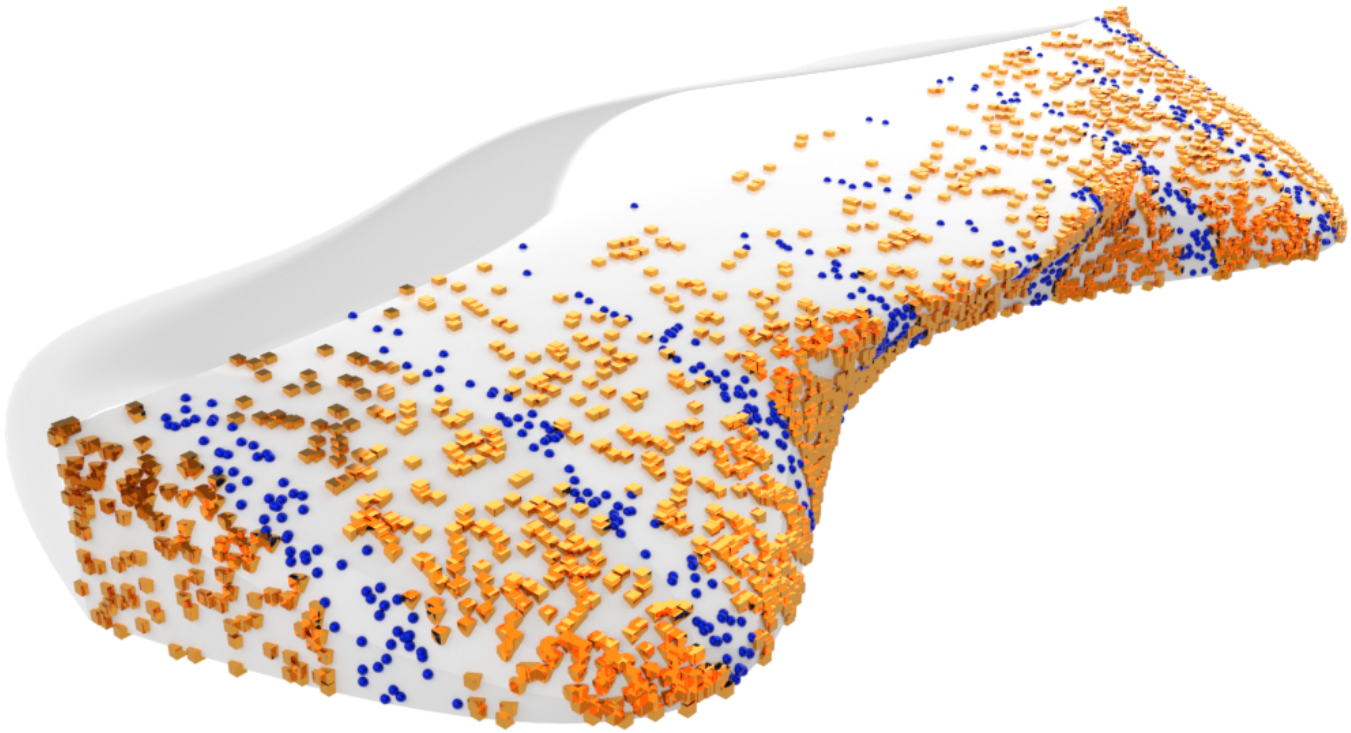

Figure 1.6: 3000 points randomly generated on the surface. The bronze cubes depict the sub-selection of on-surface points when $N = 10$ and $\lambda = 0.78$. Top view

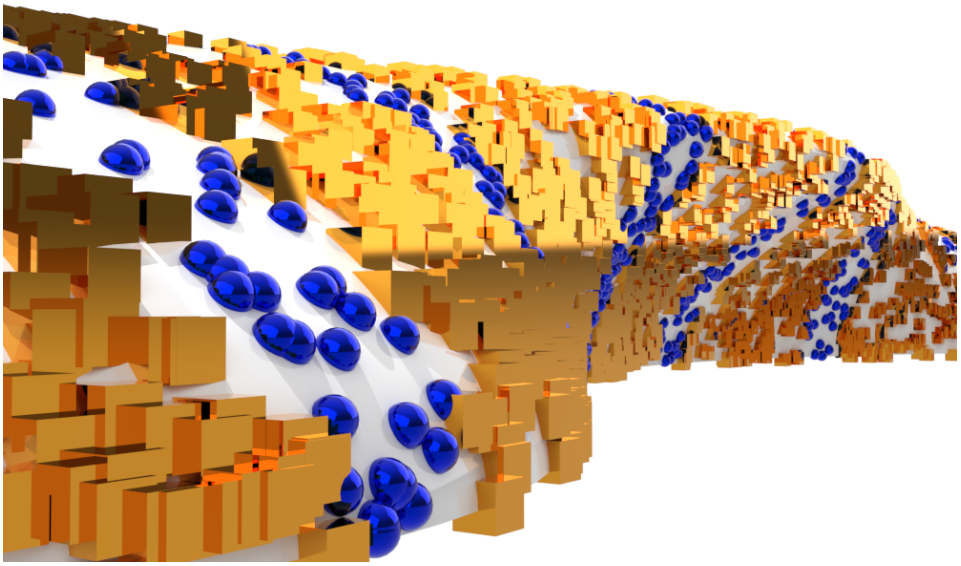

Figure 1.7: 3000 points randomly generated on the surface. The bronze cubes depict the sub-selection of on-surface points when $N = 10$ and $\lambda = 0.78$. Bottom view

Continuing to reduce λ does not further increase performance as there is not enough information (point cloud size) to do so, which clarifies the balance between deviation due to insufficient amount of information and deviation due to too much information (using points further far away from the point where the sectional area is evaluated at). See Figure 1.8 for the resulting sectional area curve with error 0.00642 if $\lambda = 0.2$ and Figure 1.9 for the on-surface selection.

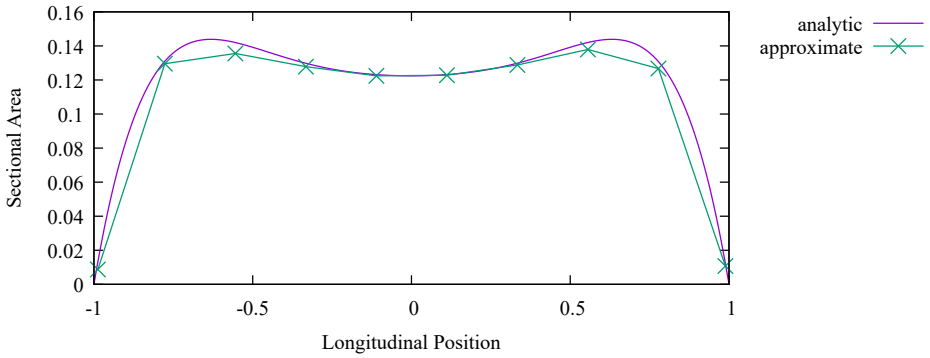

Figure 1.8: Sectional area curve at $N = 10$ points, of the geometry depicted in Figure 1.1, using 3000 randomly sampled points, and $\lambda = 0.2$. The L^∞ error at the evaluated points is 0.00642

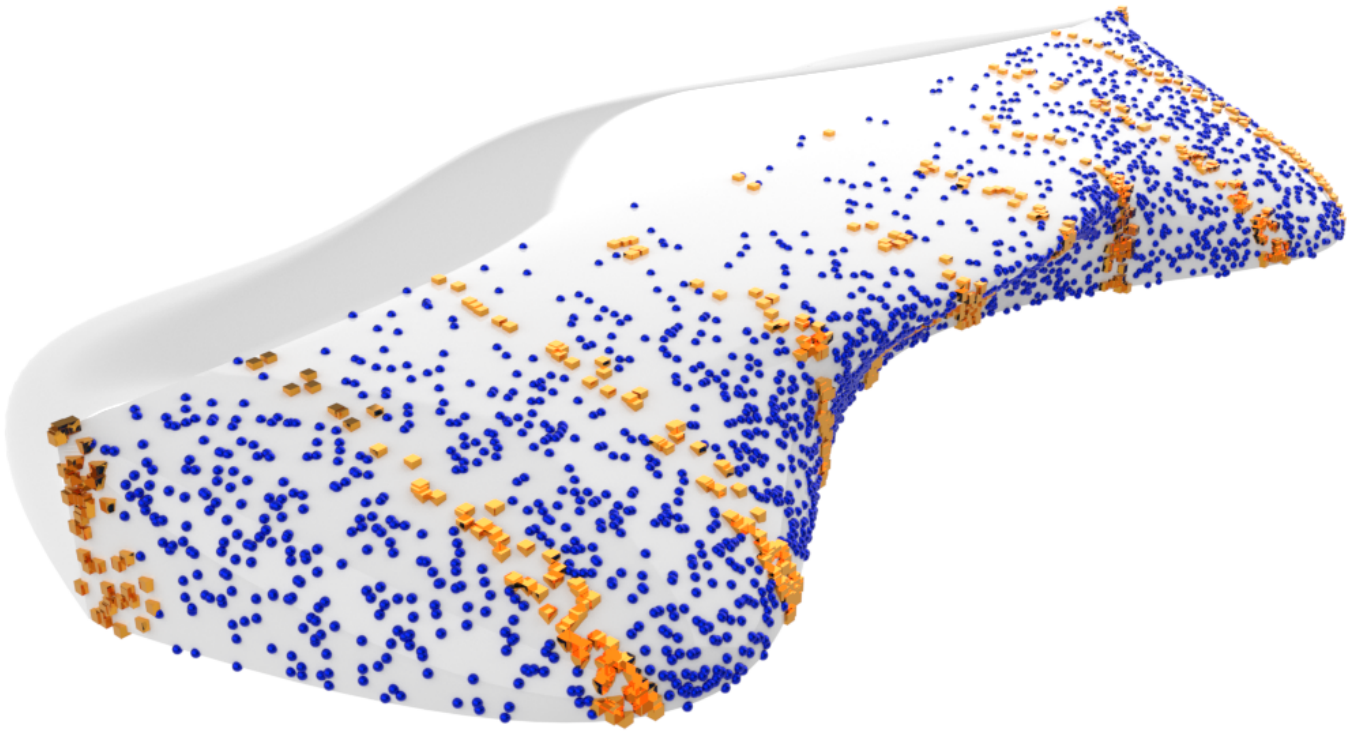

Figure 1.9: 3000 points randomly generated on the surface. The bronze cubes depict the sub-selection of on-surface points when $N = 10$ and $\lambda = 0.2$. Top view

1.2 test-2.cpp

`SectionalAreaXwiseYsymmetrical()` can be used to numerically evaluate the derivatives of the sectional area curve. For example, generating 1000 points to the aft and to the bow the extended wigley hull (see Section 1.1), then evaluating the cross sectional area at five points and using the first two calculate the 1-point finite-difference derivative, an accuracy of 0.001 relative to the analytic derivative is achieved (see Figure 1.10)

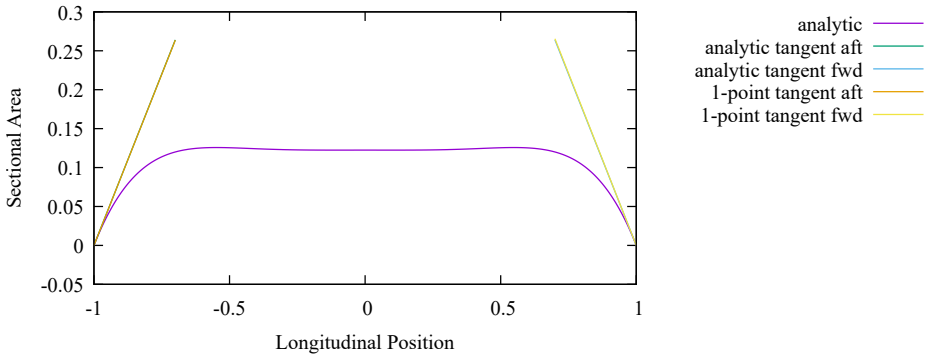

Figure 1.10: 1st derivatives of sectional area curve evaluated through 1-point finite differences using `SectionalAreaXwiseYsymmetrical()`. 1000 points were generated at aft and bow. The relative difference between the approximate and analytic derivatives is in the order of 0.001

1.3 test-3.cpp

The cross sectional area curve of the KCSsim hull (see `KCSsim.hpp` modeller and Khan et al. 2022) is calculated using `SectionalAreaXwiseYsymmetrical()` routine. Since the analytic sectional area curve of the KCSsim hull is not available, the validity of the produced curve is verified through it's integral which must be equal to the volume of the underlying geometry. Said integral is approximated numerically, while the volume is calculated through a triangulation of the KCSsim hull in 5e4 triangles.

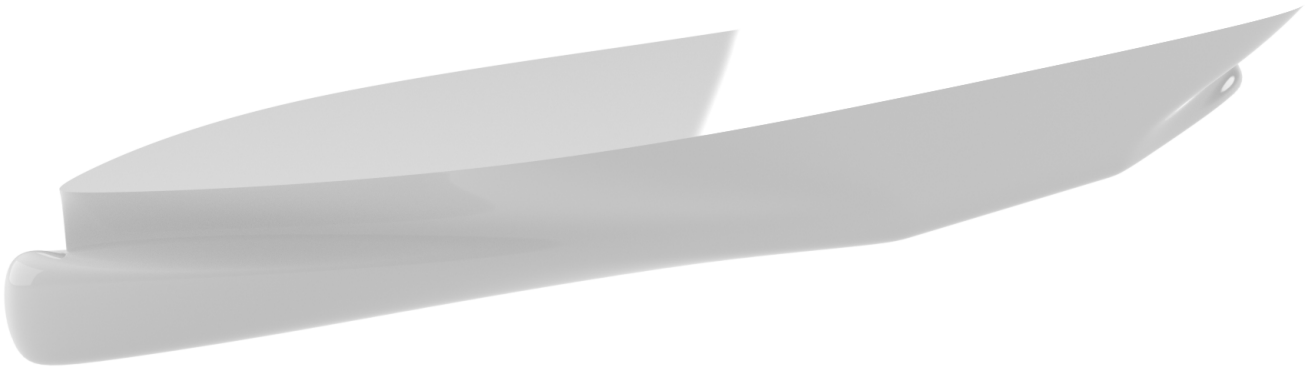

Figure 1.11: Rendered KCSsim hull. Design is documented in Table 1.1

Parameter	Value	Parameter	Value	Parameter	Value
c_1	1.0	c_{11}	0.5	c_{21}	1.0
c_2	0.8	c_{12}	0.5	c_{22}	0.0
c_3	0.8	c_{13}	0.5	c_{23}	0.5
c_4	0.265	c_{14}	0.5	c_{24}	0.5
c_5	0.543	c_{15}	0.5	c_{25}	0.5
c_6	0.0	c_{16}	0.05	c_{26}	0.5
c_7	0.3	c_{17}	0.5	c_{27}	0.5
c_8	0.65	c_{18}	0.7	c_{28}	0.5
c_9	0.7	c_{19}	0.8	c_{29}	0.5
c_{10}	0.5	c_{20}	1.0	-	-

Table 1.1: Design parameters of KCSsim hull rendered in Figure 1.11

Now, two points clouds have been generated, one random and one uniform, both having 3e4 points and the resulting sectional area curves are depicted in Figure 1.12.

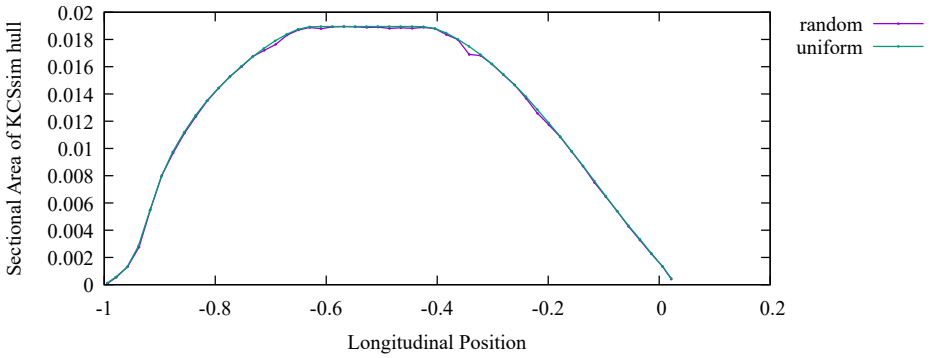

Figure 1.12: Sectional area curves from 51 sections, evaluated for the KCSsim hull documented in Table 1.1, using a random and a uniform point cloud in 3e4 points and $\lambda = 1.0$ in `SectionalAreaXwiseYsymmetrical()`. Even though the uniform sectional area curve is smoother that it's random counterpart, integrating them in order to derive the volume of the underlying geometry, yields a smaller error for the random-case. Specifically, the relative errors compared to the actual volume, for the uniformly and randomly generated point clouds were 0.01188 and 0.00564 respectively.

As is noted in Figure 1.12, the randomly-generated point cloud outperformed the uniformly generated one. The reason is that the uniformly generated point cloud was restricted to the iso-curves of the surface, thereby leaving large areas of the geometry empty, which was not the case with the randomly-generated one. This is illustrated in Figures 1.13 and 1.14, where the uniform point cloud (blue spheres), identify the contours of the surface's iso-curves, while the randomly-generated point cloud, covers the surface more evenly (ironically, more uniformly).

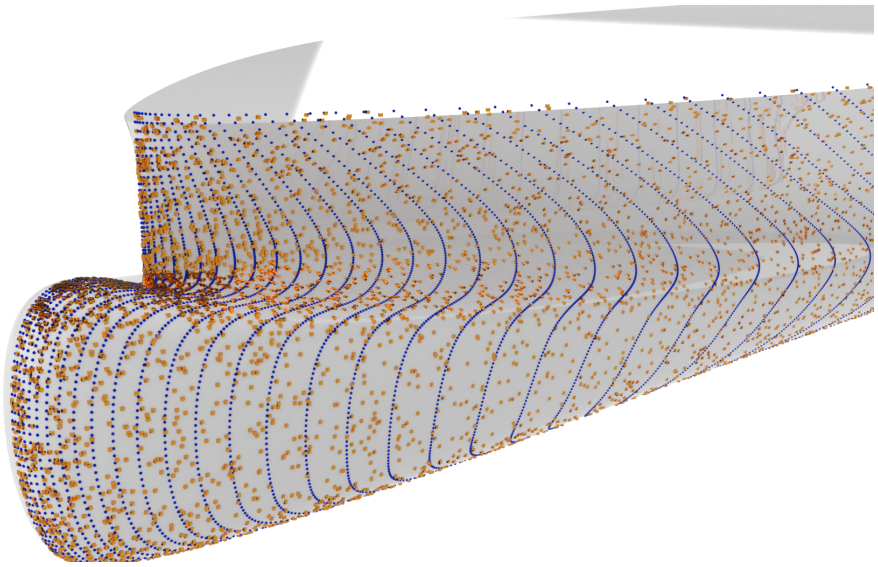

Figure 1.13: Render of bow-section of KCSsim hull documented in Table 1.1, with two point clouds: a uniformly generated point cloud (blue spheres) and a randomly generated point cloud (bronze cubes)

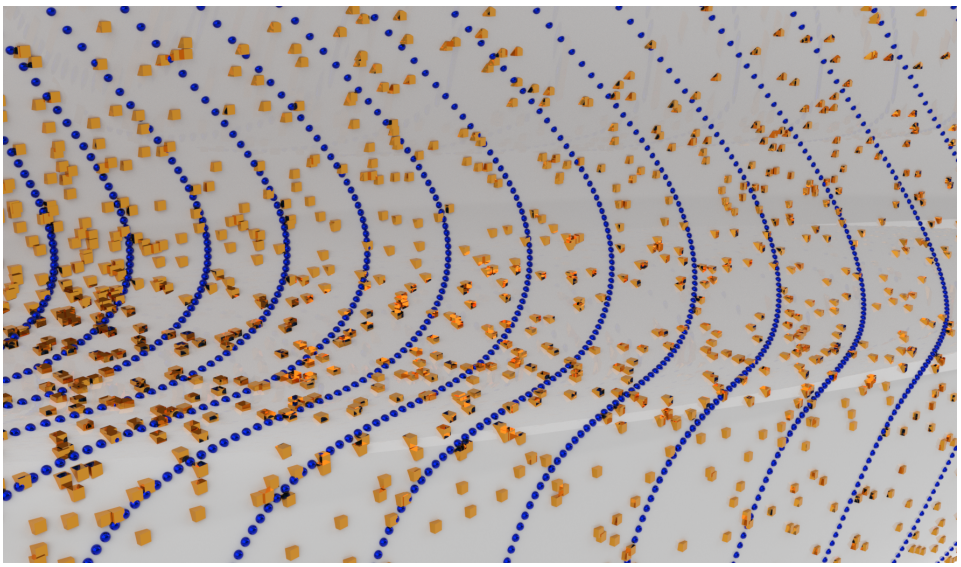

Figure 1.14: Close-up render of bow-section of KCSsim hull documented in Table 1.1, with two point clouds: a uniformly generated point cloud (blue spheres) and a randomly generated point cloud (bronze cubes)