Asociación Argentina
de Mecánica Computacional

Mecánica Computacional Vol XXXVIII, págs. 413-423 (artículo completo)
H.G. Castro, J.L. Mroginski, R.R. Paz, M.A. Storti (Eds.)
Resistencia, 1-5 Noviembre 2021

UNA HERRAMIENTA DE MODELADO PARAMÉTRICO DE FORMAS DE PESQUEROS CON PROA INVERTIDA

A PARAMETRIC SHAPE MODELER TOOL FOR INVERTED BOW FISHING VESSEL

Nicolás A. Antonelli[a,b,c], Hernán G. Kunert[a], Nicolás Biocca[b,c], Juan M. Gimenez[b,c], Gustavo E.Carr[b,c] y Santiago A. Urquiza[a,c]

[a]*Grupo HidroSim, Universidad Tecnológica Nacional, Facultad Regional Mar del Plata, Buque Pesquero Dorrego N^o 281,Mar del Plata, Argentina, email:hidrodinamica@mdp.utn.edu.ar*

[b]*CONICET - Mar del Plata, Argentina, http://mardelplata-conicet.gob.ar/*

[c]*Grupo de Ingeniería Asistida por Computadora (GIAC), Universidad Nacional de Mar del Plata, Facultad de Ingeniería, Juan B.Justo N^o 4302, Mar del Plata, Argentina, email:ingenier@fi.mdp.edu.ar*

Palabras clave: Buque, Diseño paramétrico, Fluidodinámica Computacional, Proa invertida, Hidrodinámica naval.

Resumen. La implementación de las herramientas de diseño computacionales actuales en las tareas de diseño de formas no satisfacen de manera efectiva a las exigencias de rapidez de la industria naval. Como solución a esto, surgen las técnicas de generación paramétricas de carenas a partir de geometrías base para cada tipo de buque. Por otra parte, aún no se han implementado trabajos de esta índole para el estudio buques proa invertida, tendencia que se viene utilizando cada vez más. En este trabajo se elige como entorno de trabajo la plataforma GNU Salome (https://www.salome-platform.org) y se desarrolla código en lenguaje Python para automatizar la construcción geométrica de una "familia"de cascos de pesqueros proa invertida a partir de una serie de parámetros de entrada. Se espera que los códigos implementados puedan ser integrados en futuros trabajos de optimización de formas. Se generan varios casos con variaciones de parámetros de entrada resultando en una amplia gama de alternativas de diseño.

Keywords: Ship, Parametric Design, CFD, Inverted Bow, Marine hydrodynamics.

Abstract. The implementation of current computational design tools in form design tasks does not effectively meet the demands of the shipbuilding industry. As a solution to this, there are parametric hull generation techniques that starts from base geometries for each type of ship. On the other hand, work of this nature has not yet been implemented for the study of inverted bow vessels, a trend that has been used more and more. In this work, the GNU Salome platform (https://www.salome-platform.org) is chosen as the working environment and code is developed in Python language to automate the shape construction of a inverted bow hull family from a series of input parameters. It is hoped that the implemented codes can be integrated into future shape optimization work. Several cases are generated with variations of input parameters resulting in a wide range of design alternatives.

1. INTRODUCCIÓN

Desde hace años, el diseño de los buques se basaba en diseños existentes, con determinadas modificaciones basadas en la experiencia de los profesionales, cuyo rendimiento solo era evaluable una vez construido, en canales de ensayos experimentales. No obstante, últimamente, el modelado computacional está haciendo posible calcular los rendimientos, cada vez con mayor precisión, y naturalmente, probar las variaciones geométricas, correcciones y mejoras de manera virtual, a bajo costo. Esto, sin dudas, minimiza la cantidad de ensayos en canales de experiencias físicos, de manera tal de reducir costos y tiempos.

Debido a la creciente demanda de eficiencia y robustez en el diseño asistido por computadora de buques (CASD - Computer aided ship design, por sus siglas en inglés), se les requiere a las técnicas de modelado, análisis y evaluación computacional que provean, cada vez más, mejoras medibles tanto para el proceso de diseño como para el producto resultante. Afortunadamente, gracias a la implementación de sofisticados solvers CFD y su acoplamiento con avanzadas estrategias de modelado geométrico y optimización, concretamente optimización multiobjetivo, se está llegando a productos de alto rendimiento y a su vez, de sencillo control en cuanto a parámetros, por parte del diseñador.

La optimización de la geometría del casco de un buque con respecto a minimización de resistencia al avance, de movimientos del buque, consumo de combustible, y demás, se lleva a cabo en la inmensa mayoría de los casos a partir de parametrizaciones de la geometría, bien por generación directa de una extensa lista de parámetros, o por variaciones a partir de una geometría original mediante lo que se conoce como "familia de cascos", por ejemplo, la serie 60 (Todd, 1963), para la cual hemos realizado implementaciones computacionales en Antonelli et al. (2019)

En Zhang et al. (2008) se utilizan NURBS para representar la geometría del casco a partir, tanto de parámetros locales, como globales, y generando curvas de referencia en el fondo, aristas, cubierta, y demás. En Ghassabzadeh y Ghassemi (2013) utiliza una cantidad mínima de parámetros para definir completamente la geometría de buques de planeo típicos (lanchas de velocidad), también utilizando NURBS. En Villa et al. (2020) se utilizan técnicas de reducción del espacio de diseño (reduced order models - ROMs en inglés) para generar tanto las variaciones globales como locales. Por otra parte, en Kostas et al. (2015) se realiza una parametrización de formas mediante T-Splines y una aplicación via método de Lackenby.

Por lo anteriormente expuesto, el objetivo del presente trabajo consiste en desarrollar un modelo completamente paramétrico de buques pesqueros con proa invertida, utilizando la menor cantidad de parámetros posibles. Por otra parte, se espera que los resultados sean geometrías aptas para posteriores trabajos de optimización, tanto de resistencia al avance como de movimientos del buque, principalmente.

2. PARÁMETROS GEOMÉTRICOS PRINCIPALES

Se toman 5 (cinco) puntos como referencia para trazar la linea que define el perfil del buque, esto es, la intersección de la carena con el plano de crujía. Estos son:

Punto 0: Intersección entre el fondo y la sección media. Está fijo siempre.

Punto 1: Comienzo del canto de proa (roda).

Punto 2: Fin de la roda.

Punto 3: Unión de la proa con la cubierta.

Punto 4:Intersección entre la cubierta y la sección media. Está fijo siempre.

Para definir dichos puntos, se utilizan las siguientes variables:

d_{0-1}: Distancia entre el punto 0 y el punto 1.
d_{1-2}: Distancia entre el punto 1 y el punto 2.
d_{2-3}: Distancia entre el punto 2 y el punto 3.
a_{0-1}: Ángulo comprendido entre el punto 0 y el punto 1, medido en el plano de crujía.
a_{1-2}: Ángulo comprendido entre el punto 1 y el punto 2, medido en el plano de crujía.
d_{2-3}: Ángulo comprendido entre el punto 2 y el punto 3, medido en el plano de crujía.

Las siguientes variables permanecerán constantes:
a_{1-2}: Fijado en 2 grados
a_{2-3}: Fijado de manera tal que el segmento que forman los puntos 2 y 3 sea tangente a la curvatura de la roda en el punto 2.
d_{2-3}: Fijado de manera tal que el punto 3 sea la intersección entre la proa y la cubierta, la cual siempre seguirá la misma forma (estándar, con el arrufo típico de este tipo de embarcaciones).

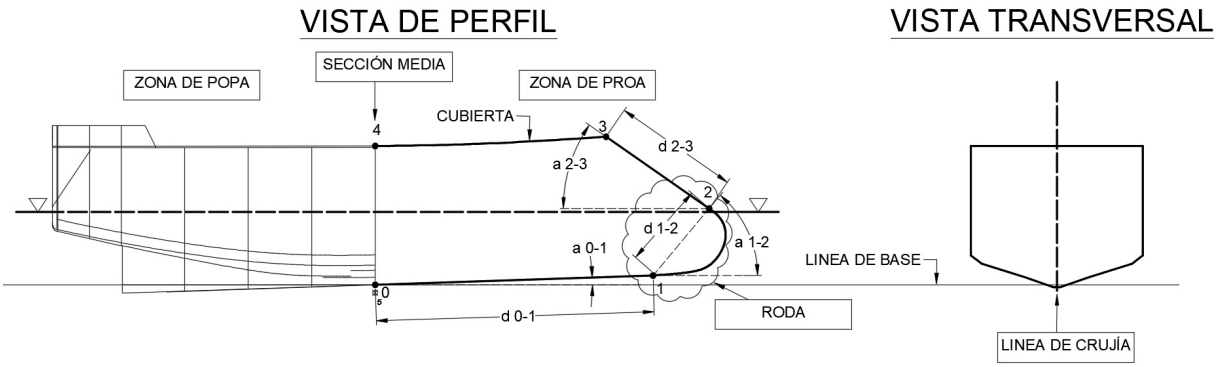

Figura 1: Descripción de los puntos y parámetros mencionados. Ver sección 3.

Ahora bien, cada miembro de la "familia de carenas"se genera a partir de los siguientes parámetros:

par1: Primer parámetro. Modifica a d 0-1. Su valor por defecto es 0, puede variar entre -1 y 1; como se verá en el siguiente apartado, está modulado con un valor estándar. Ver figura 4.

par2: Segundo parámetro. Modifica a d 1-2. Su valor por defecto es 0, puede variar entre -1 y 1; como se verá en el siguiente apartado, está modulado con un valor estándar. Ver figura 4.

par3: Tercer parámetro. Modifica a a 1-2. Su valor por defecto es 0, puede variar entre -1 y 1; como se verá en el siguiente apartado, está modulado con un valor estándar. Ver figura 4.

par4: Cuarto parámetro. Modifica la curva de la roda con una técnica tipo Morphing. Su valor por defecto es 1, puede variar entre 0 y 1; indica el grado de transformación desde la curva original (valor igual a 1) hasta una recta (valor igual a 0). Ver figura 4.

par5: Quinto parámetro. Modifica la curva de la arista inferior con una técnica tipo Morphing. Su valor por defecto es 1, puede variar entre 0 y 1; indica el grado de transformación desde la curva original (valor igual a 1) hasta una curva mas fina en proa y más convexa (valor igual a 0). Ver figura 3.

par6: Quinto parámetro. Modifica la curva de la arista superior con una técnica tipo Morphing. Su valor por defecto es 1, puede variar entre 0 y 1; indica el grado de transformación desde la curva original (valor igual a 1) hasta una curva mas fina en proa y más convexa (valor igual a 0). Ver figura 3.

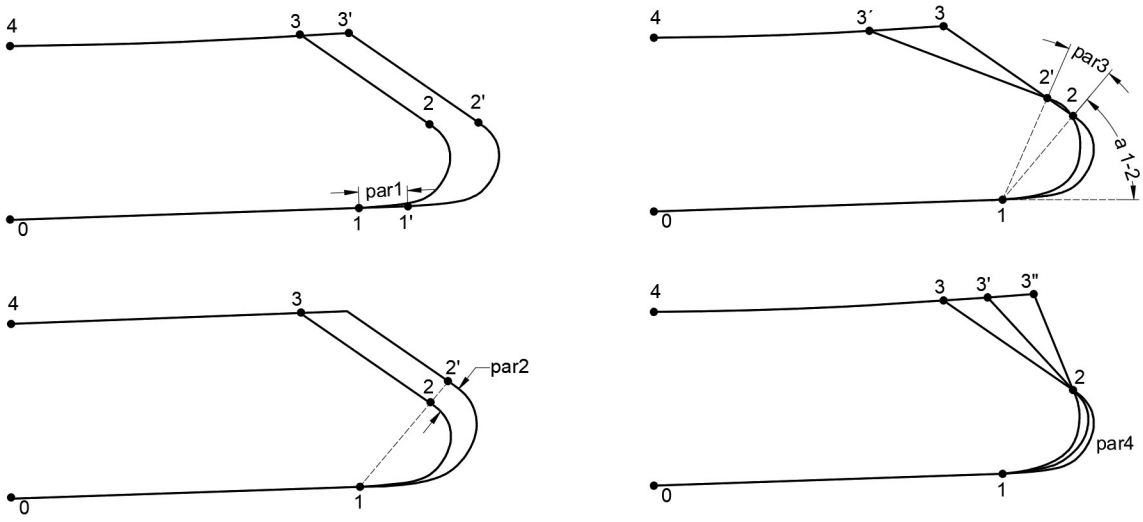

Figura 2: Descripción de los puntos y parámetros N° 1, 2, 3 y 4. Ver sección 3.

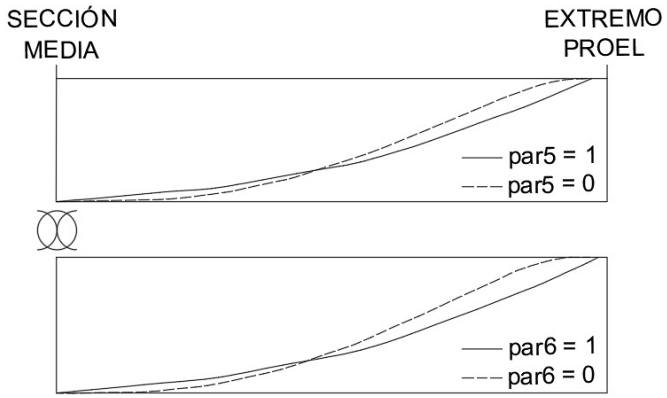

Figura 3: Descripción de los puntos y parámetros N° 5 y 6. Ver sección 3.

3. METODOLOGÍA PARA LA GENERACIÓN GEOMÉTRICA

3.1. Etapas para la representación geométrica

El casco se genera en cuatro etapas principales:

Etapa N°1: Creación de curva de crujía. En primer lugar se crea la curva de crujía, con los parámetros 1, 2, 3 y 4.

Etapa N°2: Creación de aristas. Luego se generan las aristas con los parámetros 5 y 6.

Etapa N°3: Creación de secciones transversales. Luego se generan las curvas de las secciones, a partir de 4 puntos principales más algunos auxiliares para brindar la curvatura típica de las secciones de este tipo de buques. Los puntos "principales"serían su comienzo, en el fondo, su intersección con la arista inferior, intersección con la arista superior y su finalización en la cubierta.

Etapa N°4: Creación de la superficie de la carena resultante.

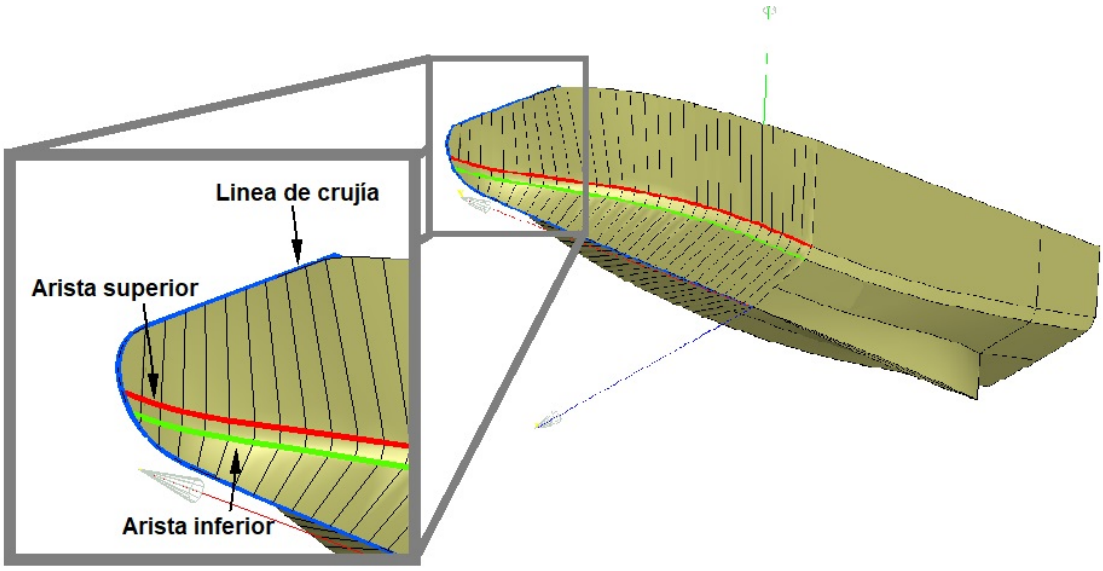

Figura 4: Descripción de los puntos y parámetros N° 1, 2, 3 y 4. Ver sección 3.

3.2. Implementacion computacional en lenguaje python

El código se estructura según el diagrama 5.

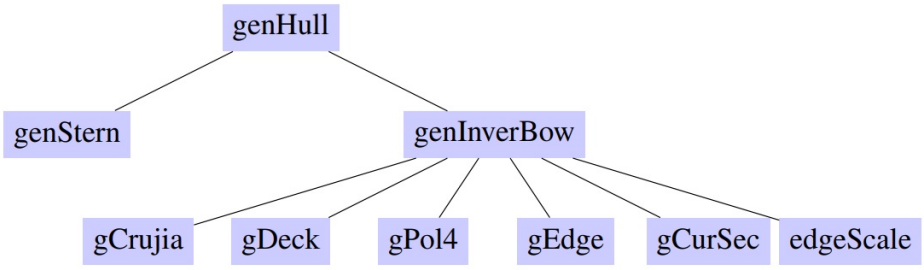

Figura 5: Estructura del código desarrollado. Ver sección 3.

El código consta de dos partes, en primer lugar una función que genera la popa y en segundo lugar, otra función que genera la proa. La primera genera siempre las mismas formas, variando únicamente las formas por delante de la sección media del buque. Al modular al buque de esta manera, se permiten futuras expansiones mediante el añadido de nuevas funciones para generar otras formas de proa.

Función genStern. Genera las secciones transversales de la popa del buque. Las mismas son constantes. No contiene subfunciones.

Función genInverBow. Depende de los 6 parametros explicados en la sección 2.

Contiene la siguientes subFunciones:

gCrujia: Genera la curva de crujia del buque a partir de los 4 parámetros descriptos anteriormente. Hasta la cubierta (punto 4)

Resulta de interés analizar su código fuente:

```
def gCrujia (par1, par2, par3, par4):
     x0 = 12.16
     y0 = 0.41
     par1_Amplitude = 0.25
     new_d01 = (1+par1*par1_Amplitude)
     x1 = x0 * new_d01 # modificacion de d01
     y1 = y0 * new_d01  # modificacion de d01
     p1 = geompy.MakeVertex(x1, y1, 0)
     geompy.addToStudy(p1, 'p1')
     curve_01 = geompy.MakeLineTwoPnt(O, p1)
     geompy.addToStudy(curve_01, 'curve_01')
```

Se indica un valor de 0.25 como máxima amplitud de la variación asociada al parámetro N°1. Se obtiene como resultado el objeto $p1$ y $curve\_01$.

```
     d12 = 4.193149 # valor original
     par2_Amplitude = 1 # amplitud maxima
     new_d12 = d12 + par2 * par2_Amplitude
     a12 = (math.pi/4)
     par3_Amplitude = (math.pi/12)
     new_a12 = a12 + par3 * par3_Amplitude
     d12_x = (math.cos(new_a12)) * new_d12
     d12_y = (math.sin(new_a12)) * new_d12
     p2 = geompy.MakeVertexWithRef(p1, d12_x, d12_y ,0)
     geompy.addToStudy(p2, 'p2')
```

Se indica un valor de 1 como máxima amplitud de la variación asociada al parámetro N°2. Se obtiene como resultado el objeto $p2$.

```
     xc4 = coordsp2[0]
     yc4 = coordsp2[1]
```

Se adopta una notación local para los 5 puntos que conformarán el canto de roda del buque. Según la figura 6, la misma comienza en el punto 1 y termina en el punto 2. Ahora, se renombra al punto 1 como punto local N°0 y al 2 como punto local N°4. Los puntos interiores, por tanto serán los N°1,2 y 3, los cuales se obtienen de una parametrización de la geometría original.

```
     curve_roda = gPol4(xc0,xc1,xc2,xc3,xc4,yc0,yc1,yc2,yc3,yc4,par4)
```

Se crea la roda a partir de los puntos descriptos anteriormente y del parámetro N°4.

```
     paux1 = geompy.MakeVertexOnCurve(Cgr4,0.9999999)
     paux2 = geompy.MakeVertexOnCurve(Cgr4,1.0000000)
```

Se crean dos puntos para trazar la tangente al final de la curva y obtener el punto 3.

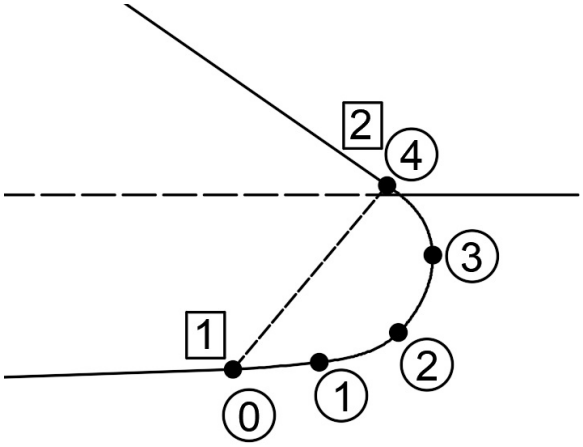

Figura 6: Descripción de la notación local utilizada para describir la roda. Los puntos en notación global están recuadrados, mientras que los puntos en notación local están en círculos

```
coordAux1 = geompy.PointCoordinates(paux1)
coordAux2 = geompy.PointCoordinates(paux2)

xPaux1 = coordAux1[0] ; xPaux2 = coordAux2[0]
yPaux1 = coordAux1[1] ; yPaux2 = coordAux2[1]

m_r2 = (yPaux1-yPaux2)/(xPaux1-xPaux2)
b_r2 = yPaux2 - m_r2 * xPaux2 # recta que pasa por dos puntos

xP4 = 0
yP4 = 6.221389

xP3 = 11.7008 # punto auxiliar
yP3 = 6.7117 # punto auxiliar

m_r1 = (yP4-yP3)/(xP4-xP3)
b_r1 = yP4 - m_r1 * xP4

p4 = geompy.MakeVertex(xP4, yP4, 0 )

xp3 = (b_r2-b_r1) / (m_r1 - m_r2)
yp3 = m_r1 * xp3 + b_r1

p3 = geompy.MakeVertex(xp3, yp3, 0 )


geompy.addToStudy(p3, 'p3')
geompy.addToStudy(p4, 'p4')
```

Se genera el punto 3 a partir del trazado de dicha recta y se hace lo propio con el punto 4.

```
cur23 = geompy.MakeLineTwoPnt(p2, p3)
```

```
geompy.addToStudy(cur23, 'cur23')

cur34 = geompy.MakeLineTwoPnt(p3, p4)
geompy.addToStudy(cur34, 'cur34')

curCrujia = geompy.MakeWire([cur01, Cgr4, cur23, cur34], 1e-07)
geompy.addToStudy(curCrujia, 'curCrujia')

return curCrujia
```

Se crean las curvas que unen los puntos 2 y 3, y 3 y 4 respectivamente. Luego se unen todas las curvas creadas en un único objeto tipo "Wire".

gDeck: Genera la curva de la cubierta (punto 3 a punto 4), de acuerdo al arrufo del buque. Es casi una linea recta, dado que se tratan de pesqueros de poca eslora.

gPol4: Genera una curva de grado 4, se utiliza para generar la curva de roda. La curva original de la roda

gEdge: Genera las aristas del buque, se utiliza para generar las aristas inferior y superior.

gCurSec: Genera las curvas de las secciones transversales del buque. Las mismas son interpolaciones utilizando B-Splines entre los puntos que conforman las secciones. Se parte desde el fondo y se construyen hacia arriba, pasando por las aristas ya generadas anteriormente. Además, a medida que se aumenta la coordenada longitudinal hasta llegar al extremo de proa, el grado de curvatura en las aristas se reduce de manera tal de lograr curvas suaves en la roda. Resulta de interés analizar su código fuente:

```
def gCurSec (xCorte, bottomEdge, topEdge,edgeLvl):
     verPlane = geompy.MakeVertex(xCorte, 3.5, 0)
     plane1 = geompy.MakePlane(verPlane, OX, 40)
     inter_curCrujia = geompy.MakeSection(plane1, curCrujia)
     [point1,point2] = geompy.ExtractShapes(inter_curCrujia, geompy.Shape
```

En primer lugar, se genera un plano seccional para intersectar las curvas principales creadas con anterioridad. En efecto, se intersecta al mismo con la curva de crujía generando dos puntos: el del fondo y el de cubierta.

```
     cpoint1 = geompy.PointCoordinates(point1)
     cpoint2 = geompy.PointCoordinates(point2)
     if (cpoint1[1]>cpoint2[1]):
          pointTop = point1
          pointBottom = point2
          cpointTop = cpoint1
          cpointBottom = cpoint2
     else:
          pointTop = point2
          pointBottom = point1
          cpointTop = cpoint2
          cpointBottom = cpoint1
```

Se identifican los puntos a partir de su coordenada vertical.

```
pointTopDeck = geompy.MakeSection(plane1, curDeck)
```

Se intersecta el plano con la curva de cubierta generando el punto de intersección entre la cubierta y el casco.

```
BE = geompy.MakeSection(plane1, bottomEdge)
TE = geompy.MakeSection(plane1, topEdge)
lis_curSec = []
lis_curSec.append(pointBottom)
lis_curSec.append(BE)
```

Se intersecta el plano con las aristas inferior y superior. Además, se crea una lista con los puntos de la sección.

```
lis_auxBE = []
for i in range (edgeLvl):
     auxBE = BE
     lis_auxBE.append(auxBE)
lis_curSec.extend(lis_auxBE)
lis_curSec.append(TE)
lis_auxTE = []
for i in range (edgeLvl):
auxTE = TE
lis_auxTE.append(auxTE)
lis_curSec.extend(lis_auxTE)
```

Se duplican los puntos de las aritas tantas veces como indice $edgeLvl$, función que disminuye linealmente al incrementar la eslora, de manera tal de ir eliminándolas suavemente.

```
if (xCorte>xMaxCubierta):
     lis_curSec.append(pointTop)
else:
     lis_curSec.append(pointTopDeck)
curSec = geompy.MakeBezier(lis_curSec, False)
geompy.addToStudy(curSec, 'curSec')
return curSec
```

Se crea la curva de la sección y se la añade al estudio.

edgeScale: Escala las coordenadas de los puntos que conformarán las aristas en función de la eslora total resultante (la cual depende de los parámetros de entrada). Al partir de una geometría original, se deben modificar las curvas que conformarán las aristas para adaptarlas a la variación de parámetros.

4. RESULTADOS

Se han generado geometrías precisas y realistas de carenas de buques de una "familia"de cascos, de manera tal que se pueden identificar fácilmente por su similitud y propiedades generales, pero a la vez distinguirse y presentar formas y características particulares, tales como el ángulo

de entrada en proa, la convexidad de las secciones, las posiciones verticales y horizontales de las aristas, etc. La geometría original con los parámetros por defecto se muestra en la figura 7.

Los códigos desarrollados, asimismo, devuelven como resultado superficies de carenas con formato tipo IGES, STEP o BREP según se requiera. Además, el investigador puede reconocer biunívocamente la geometría generada en función de los parámetros de entrada.

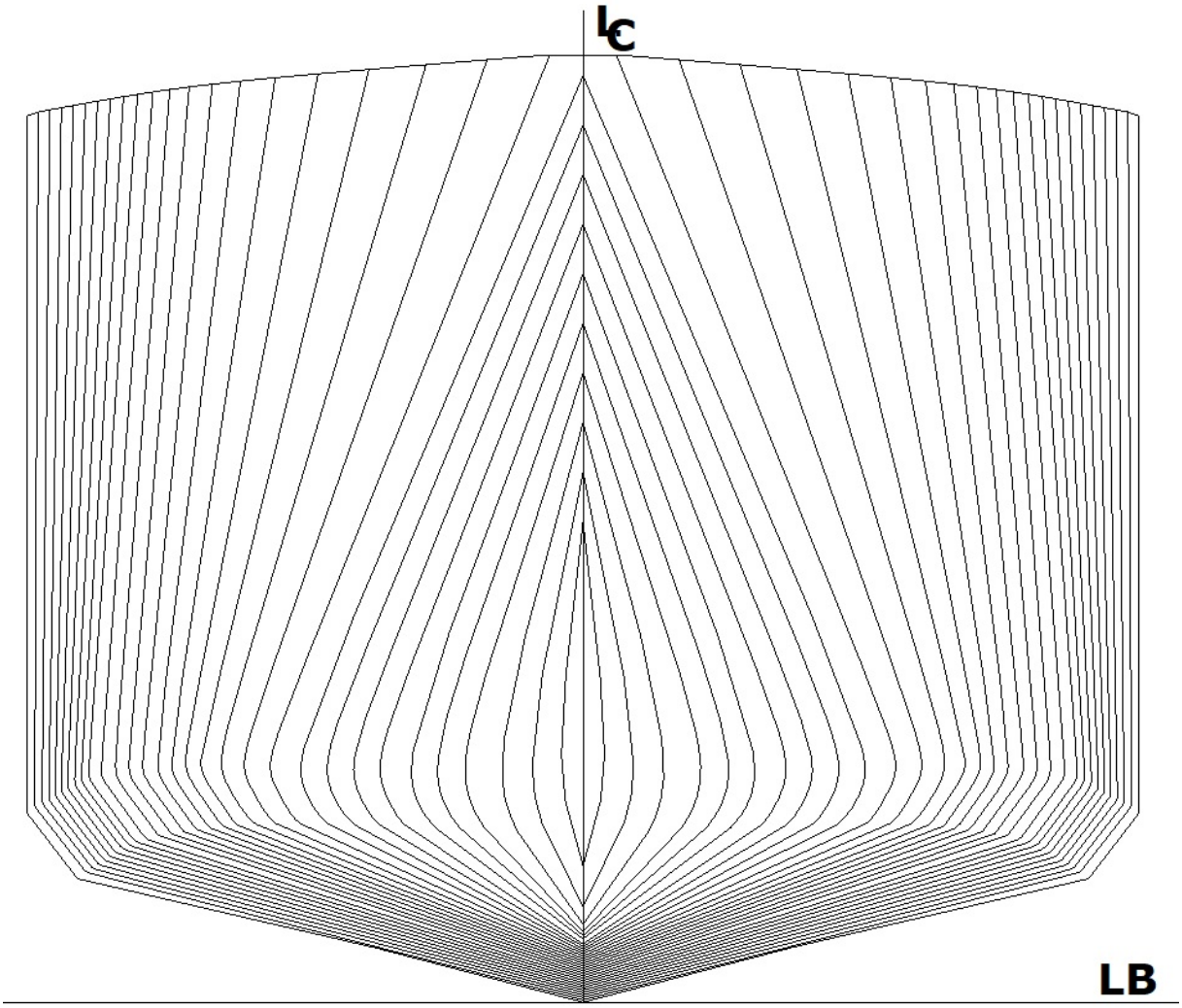

Figura 7: Secciones obtenidas con los parámetros por defecto - geometría original.

5. CONCLUSIONES

Se parte de la base de geometrías existentes para generar la popa, mientras que para la proa se construye desde cero pero manteniendo la continuidad de las aristas a partir de la escueta cantidad de 6 parámetros. Se implementó una estructura de códigos tipo script en lenguaje Python en base a funciones para cada etapa de la creación geométrica.

Se demuestra la capacidad de generar paramétricamente geometrias de pesqueros válidas para análisis CFD con una amplia gama de opciones de diseño. Además, la interfaz de programación en lenguaje Python de la plataforma Salome demostró gran potencialidad y versatilidad.

Posteriormente, se pretende integrar los códigos desarrollados dentro de un entorno que cuente con algoritmos de optimización y solvers tanto de cálculo de resistencia al avance, como de movimientos del buque.

REFERENCIAS

Antonelli N., Kunert H.G., Vaccari A., Urquiza S., Biocca N., Gimenez J.M., y Carr G. Determinación de coeficientes resistivos para buques de la serie 60 utilizando fluidodinámica computacional. *(artículo completo) Mecánica Computacional. Multiphysics*, XXXVII(43):1703–1713, 2019. ISSN 2591-3522.

Ghassabzadeh M. y Ghassemi H. An innovative method for parametric design of planing tunnel vessel hull form. *Ocean Engineering*, 60:14–27, 2013. ISSN 0029-8018. doi:https://doi.org/10.1016/j.oceaneng.2012.11.015.

Kostas K., Ginnis A., Politis C., y Kaklis P. Ship-hull shape optimization with a t-spline based bem–isogeometric solver. *Computer Methods in Applied Mechanics and Engineering*, 284:611–622, 2015. ISSN 0045-7825. doi:https://doi.org/10.1016/j.cma.2014.10.030. Isogeometric Analysis Special Issue.

Todd F. Series 60. methodical experiments with models of single-screw merchant ships. DTMB, 1963.

Villa D., Gaggero S., Coppede A., y Vernengo G. Parametric hull shape variations by reduced order model based geometric transformation. *Ocean Engineering*, 216:107826, 2020. ISSN 0029-8018. doi:https://doi.org/10.1016/j.oceaneng.2020.107826.

Zhang P., xiang Zhu D., y hao Leng W. Parametric approach to design of hull forms. *Journal of Hydrodynamics, Ser. B*, 20(6):804–810, 2008. ISSN 1001-6058. doi:https://doi.org/10.1016/S1001-6058(09)60019-6.