Práctica 2: Marco de una bicicleta

Gabryiel Bailon Avila
Carlos Eduardo Rivera López
Víctor Adrián Higuera Vázquez
Andrés Anaya Hernández
César Armando Luna Zapata

20 de Septiembre de 2022

Resumen

1. Nombre y definición de la forma

La geometría de un cuadro o marco de bicicleta a sido asociada hasta hace bien poco al concepto de antropometría (ciencia que estudia las proporciones y medidas del cuerpo humano), o lo que es lo mismo: buscar una adaptación del cuadro a las medidas corporales del ciclista[3].

2. Estado del arte

Para poder hacer o crear un buen diseño de un cuadro de bicicleta es importante saber que se dividirá en dos campos importantes, lo que es la biomecánica y la conducción[4].

2.1. Biomecánica

Todas las modalidades deportivas causan una serie de lesiones, sobre todo si se practica dicha modalidad de forma reiterada. En nuestro caso, la disciplina del ciclismo, se le atribuyen distintos tipos de lesiones como los errores en la planificación y programación de la actividad, como también lo son el gesto deportivo reiterado y basado en alteraciones morfológicas o biomecánicas. Entonces por definitiva, cualquier alteración en el acoplamiento con la bicicleta puede generar, en ligamentos y en los apoyos, tensiones y dolores anormales que constituyen posibles fuetes de inflamación, también la postura del ciclista afecta mucho, si es una postura incorrecta, esta puede producir a la larga, alteraciones en el gesto deportivo normal y producir patologías que será necesario tratar correctamente, además de emporar la rapidez y eficacia del ciclista, a continuación se mostrará una tabla con las consecuencias debido a un cambio en específico del marco de la bicicleta[1].

Defecto	Consecuencia
Cuadro de la bicicleta largo	Obliga al ciclista a estar una posición más horizontal.
Cuadro de la bicicleta corto	Obliga al ciclista a adoptar una posición más vertical, pudiendo genera lumbalgias
Cuadro de la bicicleta alto	Crea una hiperextensión de las rodillas lo que produce una sobrecarga en la musculatura posterior
Cuadro de la bicicleta bajo	Provoca una sobrecarga de la musculatura extensora
Excesiva rotación interna del pie	Sobrecarga de la musculatura externa y bíceps
Excesiva rotación externa del pie	Provoca sobrecarga de los tendones de la región interna de la rodilla y tendones de la pata de ganso
Mal alineamiento de la rótula	Dolores en la rótula, provocando condromalacia rotuliana
Altura y posición incorrecta del sillín	Dolor lumbar
Sillín muy atrás y muy alto	Provoca dolor en la rótula

Figura 1: Consecuencias para el ciclista debidas a algún cambio/modificación en el marco/cuadro de la bicicleta.

Y ahora que ya sabemos la importancia de un buen ajuste, vamos a ver qué tenemos que tener en cuenta para optimizar nuestra bicicleta en base a las medidas de nuestro cuerpo[2].

3. Propuesta del diseño de la geometría, alcances y limitaciones

1.- La talla de la bicicleta (medidas del cuadro)

Las medidas del cuadro es lo único que no podremos ajustar ya que viene dado de fábrica, por lo que es algo a considerar ANTES de la compra de la bicicleta. Nos la podemos encontrar en pulgadas (14"-23"+), centímetros (47cm-61cm+) o tallajes estándar (S, M, L, XL...). Además dependerá del tipo de bicicleta (montaña, carretera, paseo...) y del sexo.

¿Cómo se calcula?

Para bici de montaña (MTB):

Talla en pulgadas = Altura entrepierna x 0,21

Talla en centímetros = Altura entrepierna x 0,54

Para bici de carretera

Talla en centímetros = Altura entrepierna x 0,65

Con las medidas que obtengamos debemos ir a los datos del fabricante para ver qué talla se queda más cerca del resultado obtenido. En la mayoría de las webs de las distintas marcas encontraremos tablas de equivalencia de tallas pero si no encuentras nada puede usar la siguiente referencia:

Talla genérica	Talla en centímetros	Talla en pulgadas
XS	47-49	14-15
S	49-53	15-17
M	53-57	17-19
L	57-61	19-21
XL	+61	+22

Figura 2: Si tu resultado se queda justo en medio de dos tallas lo ideal es irse a la inferior, ya que ajustando el sillín (como veremos más adelante) o con una potencia más larga podremos conseguir ese extra de tamaño necesario.

2.- Altura del sillín

Importantísimo y muy pasado por alto por un gran número de ciclistas. Un buen indicador de si lo llevamos mal ajustado es el siguiente:

Sillín alto: Molestias en la parte posterior de las rodillas. Movimiento excesivo de caderas (caderéo).

Sillín bajo: Molestias en la parte delantera de las rodillas.

¿Cómo se calcula?

Altura del sillín = Medida de nuestra entrepierna x 0,88

Figura 3: La altura del sillín se mide desde el centro del eje de pedalier siguiendo la línea del tubo del sillín (no verticalmente) hasta la parte superior del sillín.

3.- Desplazamiento horizontal del sillín (retroceso)

Si la altura del sillín es algo que se pasa por alto, el desplazamiento longitudinal del sillín directamente o no se hace o se hace a ojo la mayoría de los casos. Bien, si te estabas preguntando para qué era la cuchara ahora lo sabrás. A no ser que tengas una plomada en casa vamos a tener que fabricarnos una casera. Para ello solo tienes que atar la cuchara a la cuerda y plomada al instante. Ahora toca subirse a la bicicleta y colocar los pedales alineados en posición horizontal de forma que las bielas queden paralelas al suelo. Es importante que nos sentemos correctamente, dejando caer los isquiones sobre la parte ancha del sillín. Para esto mejor que alguien te sujete la bicicleta porque te puedes ir al suelo fácilmente.

¿Cómo se ajusta?

Cogeremos nuestra plomada casera y la colocaremos justo por delante de la rodilla de forma que la plomada caiga por debajo de la biela.

Figura 4: El ajuste perfecto es el que hará coincidir la cuerda de la plomada con el extremo de la biela (no del pedal).

4. Procedimiento de la programación

Por medio de MatLab implementarremos y modificaremos el código de optimización de topología de 99 líneas utilizado en la práctica uno en el espacio que se generó al realizar la práctica pasada.

```
%%% A 99 LINE TOPOLOGY OPTIMIZATION CODE BY OLE SIGMUND, JANUARY 2000 %%%
%%% CODE MODIFIED FOR INCREASED SPEED, September 2002, BY OLE SIGMUND %%%
function topl(nelx,nely,volfrac,penal,rmin)
nelx=60;
nely=20;
volfrac=0.5;
penal=3.0;
rmin=1.5;
% INITIALIZE
x(1:nely,1:nelx) = volfrac;
loop = 0;
change = 1.;
% START ITERATION
while change > 0.01
loop = loop + 1;
xold = x;
% FE-ANALYSIS
[U]=FE(nelx,nely,x,penal);
% OBJECTIVE FUNCTION AND SENSITIVITY ANALYSIS
[KE] = lk;
c = 0.;
for ely = 1:nely
for elx = 1:nelx
n1 = (nely+1)*(elx-1)+ely;
n2 = (nely+1)* elx +ely;
Ue = U([2*n1-1;2*n1; 2*n2-1;2*n2; 2*n2+1; 2*n2+2; 2*n1+1;2*n1+2],1);
c = c + x(ely,elx)^penal*Ue'*KE*Ue;
```

Figura 5: Código de la práctica 1.

Se realizaron las siguientes modificaciones:

-Se modificaron los parámetros nelx, nely, volfrac, penal y rmin para realizar la optimización que se mostrará en la figura, donde la funcion de cada uno de estos está dada por:

nelx: Es el número de elementos finitos en el eje x.

nely: Es el número de elementos finitos en el eje y.

volfrac: Corresponde a la fracción de volumen en el dominio de diseño.

penal: Es la penalización de las densidades intermedias. En odne una penalización considerada como alta hará la solución en blanco y negro, por lo tanto los elementos finitos estarán llenos o vacíos,en cambio, una penalización igual a 1 significa que no hay penalización de las densidades intermedias.

rmin: Es el radio de filtro que hace el diseño de malla-independiente. Una vez conocidas las funciones de cada uno, estos se implementaron de la siguiente manera:

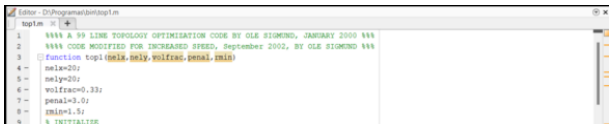


Figura 6: Modificación de los parámetros nelx, nely, volfrac, penal y rmin de la práctica 2.

Antes de iniciar la iteración, entre las líneas 12 a 21 se le agregó al código las líneas mostradas de la figura 5, se realizó debido a que no se deja un espacio libre para el marco de la bicicleta, para poder hacer este espacio se implementó una matriz con ceros.

Figura 7: Líneas de código 12 a 21 de la práctica 2.

Posteriormente, se modificó la línea 44 para realizar la optimización topológica de forma correcta y poder modificar las líneas 46, 54 y 58 del código.

Figura 8: Línea de código 44 de la práctica 2.

Se definió el tamaño de los elementos finitos a una vez por una unidad en la línea 46, esto se hace para realizar la correcta optimización topológica, de esta manera es posible ejecutar las correcciones de escala.

Figura 9: Línea de código 46 que define el tamaño de los elementos de la práctica 2.

Se modificaron las líneas 54 y 58 añadiendo el comando que empieza los elementos de la zona vacía con el valor de 0.001 que corresponde a xnew(find(passive))=0.001 para realizar la optimización topológica.

```
52 -    end
53      %%%%%%%%%% OPTIMALITY CRITERIA UPDATE %%%%%%%%%
54      function [xnew]=OC(nelx,nely,x,volfrac,dc,passive)
55 -    l1 = 0; l2 = 100000; move = 0.2;
56 -    while ((l2-l1)/l2 > 1e-4)
57 -    lmid = 0.5*(l2+l1);
58 -    xnew(find(passive))=0.001
59 -    xnew = max(0.001,max(x-move,min(1.,min(x+move,x.*sqrt(-dc./lmid)))));
60 -    if sum(sum(xnew)) - volfrac*nelx*nely > 0;
61 -    l1 = lmid;
62 -    else
63 -    l2 = lmid;
64 -    end
65 -    end
66      %%%%%%%%%% MESH-INDEPENDENCY FILTER %%%%%%%%%%
```

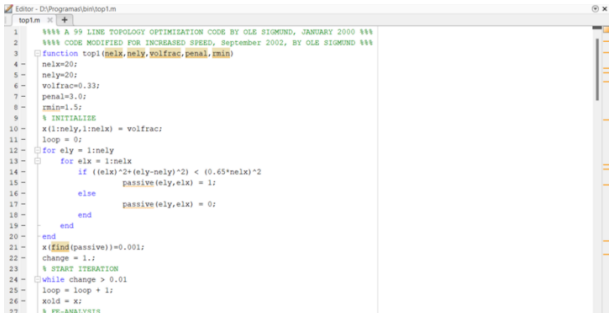
Figura 10: Modificación en las líneas de código 54 y 58 de la práctica 2.

Por medio de las líneas 86 y 87 del código se considera solo la carga y el apoyo de la bicicleta para la optimización topológica.

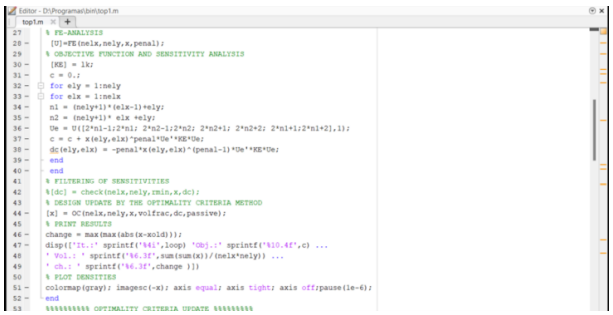
```
81 -    edof = [2*n1-1; 2*n1; 2*n2-1; 2*n2; 2*n2+1;2*n2+2;2*n1+1; 2*n1+2];
82 -    K(edof,edof) = K(edof,edof) + x(ely,elx)^penal*KE;
83 -    end
84 -    end
85      % DEFINE LOADSAND SUPPORTS(HALF MBB-BEAM)
86 -    F(2,1) = 1;
87 -    fixeddofs =2*nelx*(nely+1)+1:2*(nelx+1)*(nely + 1);
88 -    alldofs = [1:2*(nely+1)*(nelx+1)];
89 -    freedofs = setdiff(alldofs,fixeddofs);
90      % SOLVING 127
91 -    U(freedofs,:) = K(freedofs,freedofs) \F(freedofs,:);
92 -    U(fixeddofs,:)= 0;
93      %%%%%%%%%% ELEMENT STIFFNESS MATRIX %%%%%%%
```

Figura 11: Definición de la optimización topológica por medio de las líneas 86 y 87 de la práctica 2.

Código completo:



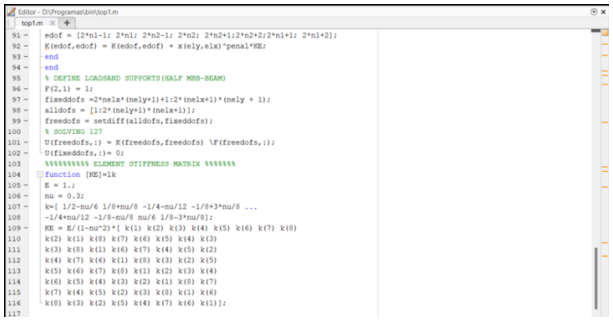
```
1       %%% A 99 LINE TOPOLOGY OPTIMIZATION CODE BY OLE SIGMUND, JANUARY 2000 %%%
2       %%% CODE MODIFIED FOR INCREASED SPEED, September 2002, BY OLE SIGMUND %%%
3       function top1(nelx,nely,volfrac,penal,rmin)
4 -     nelx=20;
5 -     nely=20;
6 -     volfrac=0.33;
7 -     penal=3.0;
8 -     rmin=1.5;
9       % INITIALIZE
10 -    x(1:nely,1:nelx) = volfrac;
11 -    loop = 0;
12 -    for ely = 1:nely
13 -        for elx = 1:nelx
14 -            if ((elx)^2+(ely-nely)^2) < (0.65*nelx)^2
15 -                passive(ely,elx) = 1;
16 -            else
17 -                passive(ely,elx) = 0;
18 -            end
19 -        end
20 -    end
21 -    x(find(passive))=0.001;
22 -    change = 1.;
23      % START ITERATION
24 -    while change > 0.01
25 -    loop = loop + 1;
26 -    xold = x;
```



```
27      % FE-ANALYSIS
28 -     [U]=FE(nelx,nely,x,penal);
29      % OBJECTIVE FUNCTION AND SENSITIVITY ANALYSIS
30 -     [KE] = lk;
31 -     c = 0.;
32 -     for ely = 1:nely
33 -     for elx = 1:nelx
34 -     n1 = (nely+1)*(elx-1)+ely;
35 -     n2 = (nely+1)* elx +ely;
36 -     Ue = U([2*n1-1;2*n1; 2*n2-1;2*n2; 2*n2+1; 2*n2+2; 2*n1+1;2*n1+2],1);
37 -     c = c + x(ely,elx)^penal*Ue'*KE*Ue;
38 -     dc(ely,elx) = -penal*x(ely,elx)^(penal-1)*Ue'*KE*Ue;
39 -     end
40 -     end
41      % FILTERING OF SENSITIVITIES
42      %[dc] = check(nelx,nely,rmin,x,dc);
43      % DESIGN UPDATE BY THE OPTIMALITY CRITERIA METHOD
44 -     [x] = OC(nelx,nely,x,volfrac,dc,passive);
45      % PRINT RESULTS
46 -     change = max(max(abs(x-xold)));
47 -     disp(['It.:' sprintf('%4i',loop) 'Obj.:' sprintf('%10.4f',c) ...
48      ' Vol.: ' sprintf('%6.3f',sum(sum(x))/(nelx*nely)) ...
49      ' ch.: ' sprintf('%6.3f',change )])
50      % PLOT DENSITIES
51 -     colormap(gray); imagesc(-x); axis equal; axis tight; axis off;pause(1e-6);
52 -     end
53      %%%%%%%%%% OPTIMALITY CRITERIA UPDATE %%%%%%%%%
```

```
%%%%%%%%%% OPTIMALITY CRITERIA UPDATE %%%%%%%%%
function [xnew]=OC(nelx,nely,x,volfrac,dc,passive)
l1 = 0; l2 = 100000; move = 0.2;
while ((l2-l1)/l2 > 1e-4)
lmid = 0.5*(l2+l1);
xnew(find(passive))=0.001
xnew = max(0.001,max(x-move,min(1.,min(x+move,x.*sqrt(-dc./lmid)))));
if sum(sum(xnew)) - volfrac*nelx*nely > 0;
l1 = lmid;
else
l2 = lmid;
end
end
%%%%%%%%%% MESH-INDEPENDENCY FILTER %%%%%%%%%%
function [dcn]=check(nelx,nely,rmin,x,dc)
dcn=zeros(nely,nelx);
for i = 1:nelx
for j = 1:nely
sum=0.0;
for k = max(i-round(rmin),1):min(i+round(rmin),nelx)
for l = max(j-round(rmin),1):min(j+round(rmin), nely)
fac = rmin-sqrt((i-k)^2+(j-l)^2);
sum = sum+max(0,fac);
dcn(j,i) = dcn(j,i) + max(0,fac)*x(l,k)*dc(l,k);
end
end
dcn(j,i) = dcn(j,i)/(x(j,i)*sum);
```

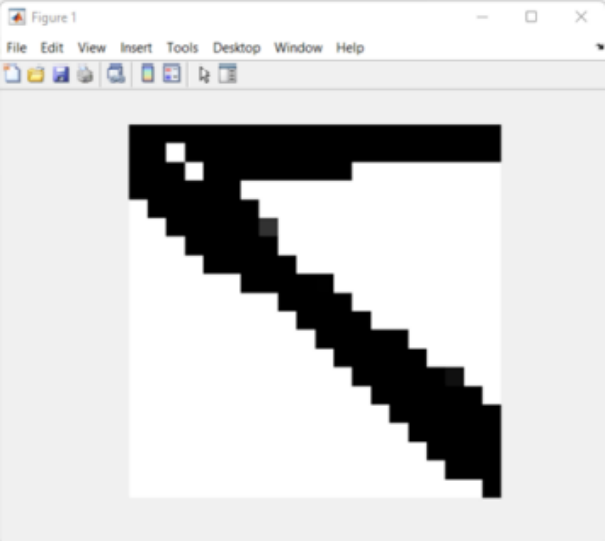
```
dcn(j,i) = dcn(j,i)/(x(j,i)*sum);
end
end
%%%%%%%%%% FE-ANALYSIS %%%%%%%%%%
function [U]=FE(nelx,nely,x,penal)
[KE] = lk;
K = sparse(2*(nelx+1)*(nely+1), 2*(nelx+1)*(nely+1));
F = sparse(2*(nely+1)*(nelx+1),1); U =sparse(2*(nely+1)*(nelx+1),1);
for ely = 1:nely
for elx = 1:nelx
n1 = (nely+1)*(elx-1)+ely;
n2 = (nely+1)* elx +ely;
edof = [2*n1-1; 2*n1; 2*n2-1; 2*n2; 2*n2+1;2*n2+2;2*n1+1; 2*n1+2];
K(edof,edof) = K(edof,edof) + x(ely,elx)^penal*KE;
end
end
% DEFINE LOADSAND SUPPORTS(HALF MBB-BEAM)
F(2,1) = 1;
fixeddofs =2*nelx*(nely+1)+1:2*(nelx+1)*(nely + 1);
alldofs = [1:2*(nely+1)*(nelx+1)];
freedofs = setdiff(alldofs,fixeddofs);
% SOLVING 127
U(freedofs,:) = K(freedofs,freedofs) \F(freedofs,:);
U(fixeddofs,:)= 0;
%%%%%%%%%% ELEMENT STIFFNESS MATRIX %%%%%%%
function [KE]=lk
E = 1.;
```



```
edof = [2*n1-1; 2*n1; 2*n2-1; 2*n2; 2*n2+1;2*n2+2;2*n1+1; 2*n1+2];
K(edof,edof) = K(edof,edof) + x(ely,elx)^penal*KE;
end
end
% DEFINE LOADSAND SUPPORTS(HALF MBB-BEAM)
F(2,1) = 1;
fixeddofs =2*nelx*(nely+1)+1:2*(nelx+1)*(nely + 1);
alldofs = [1:2*(nely+1)*(nelx+1)];
freedofs = setdiff(alldofs,fixeddofs);
% SOLVING 127
U(freedofs,:) = K(freedofs,freedofs) \F(freedofs,:);
U(fixeddofs,:)= 0;
%%%%%%%%%% ELEMENT STIFFNESS MATRIX %%%%%%%
function [KE]=lk
E = 1.;
nu = 0.3;
k=[ 1/2-nu/6 1/8+nu/8 -1/4-nu/12 -1/8+3*nu/8 ...
-1/4+nu/12 -1/8-nu/8 nu/6 1/8-3*nu/8];
KE = E/(1-nu^2)*[ k(1) k(2) k(3) k(4) k(5) k(6) k(7) k(8)
k(2) k(1) k(8) k(7) k(6) k(5) k(4) k(3)
k(3) k(8) k(1) k(6) k(7) k(4) k(5) k(2)
k(4) k(7) k(6) k(1) k(8) k(3) k(2) k(5)
k(5) k(6) k(7) k(8) k(1) k(2) k(3) k(4)
k(6) k(5) k(4) k(3) k(2) k(1) k(8) k(7)
k(7) k(4) k(5) k(2) k(3) k(8) k(1) k(6)
k(8) k(3) k(2) k(5) k(4) k(7) k(6) k(1)];
```

Una vez escrito todo el código, se guardó, prestando atención en la ubicación donde se guardó el script, así como en el nombre que se le asignó al archivo. En este caso, otra vez se le guardó con el nombre de top1, donde su extensión por default es de .m, y finalmente, se ejecutó el programa haciendo uso del botón de “Run”.

5. Optimización



6. Conclusiones

César Armando Luna Zapata 1844920 IMTC

Como conclusión en la práctica dos del laboratorio, una vez comprendido cómo funciona la optimización topológica, por medio de la modificación del código de 99 líneas de la práctica uno, fue posible realizar la aplicación de este a un marco de una bicicleta, solo considerando para esta práctica la carga y el apoyo de la misma, realizando cambios en los parámetros del código, principalmente en nelx, nely, volfrac, penal y rmin donde nelx es el número de elementos finitos en el eje x, nely es el número de elementos finitos en y, volfrac es la fracción de volumen en el dominio de diseño, penal se encarga de la penalización de las densidades intermedias y por último, rmin es un radio de filtro que hace el diseño de malla-independiente. Es importante destacar que la optimización topológica es una gran herramienta para el diseño, asimismo, programas como Matlab nos ayudan a realizar las optimizaciones de una manera sencilla y eficaz, permitiendo obtener grandes conocimientos de aplicación en el laboratorio de biomecánica.

Gabryiel Bailon Avila 1869828 IMTC

Esta segunda práctica hizo uso del código de optimización topológica implementado en la práctica anterior, es este caso, para poder desarrollar el marco de una bicicleta, es gracias a esto, que el desarrollo del código mostrado con anterioridad, resultó ser bastante sencillo tanto de utilizar, así como de comprender su funcionamiento. Uno de los cambios más sobresalientes realizados al código original, fueron los que se hicieron en los parámetros nelx, nely, volfrac, penal y rmin, esto con el objetivo de llevar a cabo la optimización que se obtuvo como resultado. A su vez, se modificaron diversas líneas del código, donde cabe destacar el tamaño de los elementos finitos a una vez por una unidad en la línea 46, lo cual se hizo para poder realizar correctamente la optimización, puesto que de esta manera, fue posible ejecutar las correcciones de escala. Finalmente, por medio de las líneas 86 y 87 del código, se considera única y exclusivamente la carga y el apoyo de la bicicleta para la optimización.

Víctor Adrián Higuera Vázquez 1876474 IMTC

Para esta práctica se realizó un código de optimización topológica, con esto se puede utilizar para una amplia investigación en nuestro caso fue desarrollar un marco de bicicleta para implementar un mejor diseño y que sea 100tanto en estética como en peso y aerodinámico. Sin duda una tarea algo compleja pero con lo realizado en la práctica anterior se complementó.

Carlos Eduardo Rivera López 1897545 IMTC

En esta segunda utilizamos lo que fue la optimización topológica que vimos en la práctica 11, pero ahora de manera

más implementaría, para poder desarrollar un diseño de marco para una bicicleta, generando un diseño que tuviere la mejor resistencia y optimización posible, fuera de la modificación de algunas partes del código, la práctica no contó con una dificultad muy elevada, lo que hizo que aprender de esta práctica y comprender cómo funcionaba fuera más fácil, por lo que este tema quedó comprendido muy bien y es de un gran apoyo.

Andrés Anaya Hernández 1914471 IMTC

En esta segunda práctica, comprendimos y analizamos más a detalle el proceso de optimización topológica, todo esto aplicado a una propuesta de análisis y modificación a un marco de bicicleta como es mencionado en el nombre de la actividad. Todo esto se vio plasmado en la realización del código en la aplicación matlab misma que nos permitió apreciar todo esto en forma gráfica. Indagamos un poco en nuevos comandos de modificación como lo fueron nel x y y, penal, entre otros. A lo largo del desarrollo de la práctica, lo más destacable fue el tener cada detalle en orden en las distintas líneas de código, mismas que nos arrojaron un resultado satisfactorio.

Referencias

[1]

[2]

[3] José Carlos Pedrero. GeometrÍa de un cuadro de bicicleta, Noviembre 2018. URL `https://labicicleta.net/escuela/geometria-de-un-cuadro-de-bicicleta/`.

[4] Tesis y Masters. ¿cómo se hace el estado del arte de una tesis?, 2021. URL `https://tesisymasters.com.ar/como-se-hace-el-estado-del-arte-de-una-tesis/`.