Universidad Autónoma de Nuevo León

Facultad de Ingeniería Mecánica y Eléctrica

Práctica #3: Diseño de la estructura de un panorámico

Equipo 10

Datos del equipo:

Nombre	**Matrícula**	**Carrera**
Andrés Anaya Hernández	**1914471**	IMC
Carlos Eduardo Rivera López	**1897545**	IMC
César Armando Luna	**1844920**	IMC
Víctor Adrián Higuera Vázquez	**1876474**	IMC
Gabryiel Bailon Avila	**1869828**	IMC

Catedrático: M.C. Yadira Moreno Vera

Fecha de entrega: 01 de Noviembre de 2022

Hora: V2 Brigada: 214

Semestre: Agosto - Diciembre 2022

Práctica #3: Diseño de la estructura de un panorámico.

Objetivo:

El estudiante deberá presentar un reporte con la solución numérica computacional del problema de la simulación del desempeño mecánico de componentes mecánicos por medio del método de MATLAB, esto para desarrollar en el estudiante la capacidad de análisis, implementación y solución de un problema propuesto.

Estado del arte

La necesidad de orientación espacial ha estado con el hombre desde el principio de los tiempos. Saber dónde estamos, qué hacer, hacia donde ir, son actividades que realizamos cotidianamente. A través del tiempo se han creado diversas formas de marcar lugares y ofrecer indicaciones. Desde el imperio romano (S. II a IV D. C.) se marcaban los caminos para llegar a Roma con columnas cada milla, por esto eran llamados miliarios romanos, y contenían la información de la distancia e información de gobernantes y localidades aledañas. Ya en la edad media se marcaban las encrucijadas y caminos principales con tablas o placas metálicas. En Alemania en el siglo XV, era obligatorio para los locales donde se vendía cerveza colocar un letrero para denotar su carácter comercial.

A lo largo del tiempo las señales fueron más necesarias para indicar lugares y direcciones, existen múltiples registros de marcas en caminos de piedras, tablas y carteles para orientar a los transeúntes. Pero fue hasta principios del siglo XX y con el advenimiento del invento del automóvil que se hizo necesario crear y homologar un sistema señalético para carreteras. Dicha homologación se efectúo en 1908 en el Primer Congreso Internacional de Tránsito en Roma y estandarizó las señales viales, prácticamente sin cambio hasta nuestros días

Estas estructuras usualmente se encuentran en medio de diversos paisajes urbanos y sostienen diseños publicitarios con el objetivo de promocionar un producto, servicio o transmitir un mensaje.
Cada país tiene ciertas normativas en cuanto a dónde es apropiado o no colocar estos soportes para anuncios publicitarios. En algunos no está permitido que se construyan estructuras panorámicas a loslados de autopistas porque estos pueden distraer a los conductores.
Los panorámicos se exponen a altas ráfagas de viento, por lo que su estructura ocupa ser muy rígidapara soportar estas fuerzas.

Propuesta de diseño de la geometría, alcances y limitaciones

En la figura 1 se muestra el panorámico que será el espacio de diseño a evaluar, éste será de 2 dimensiones, con cargas y apoyos como se muestra a continuación:

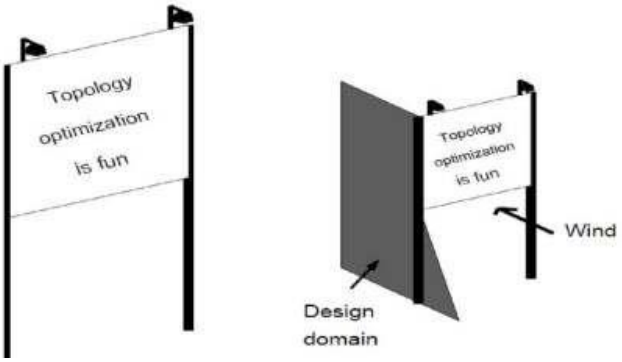


Figura 1. Imagen del panorámico.

Existen diversos materiales que las agencias especializadas usan en la creación y diseño para estas estructuras. Usualmente los postes panorámicos están construidos de metal y acero para que sean lo suficientemente resistentes al clima, lluvias y cualquier otro fenómeno de la naturaleza.
Por otra parte, el panorámico en sí mismo son hechos de lona, vallas de PVC, plástico, tela, metal o acrílico. También existen espectaculares digitales o electrónicos que tienen luces, pantallas eléctricas y música.
En la figura 2 se puede ver el espacio de diseño para esta práctica. Se espera una fracción volumétrica aproximada de 0.20% del espacio de diseño. Supongamos que el panorámico es muy rígido 1, y sus patas son del mismo material que el marco.

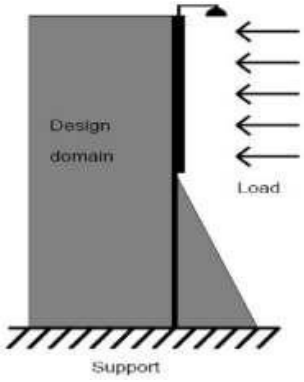
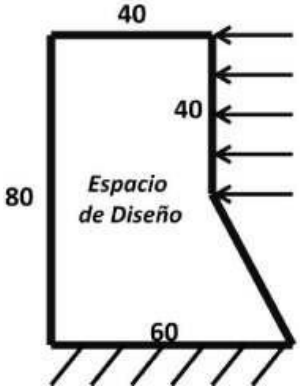


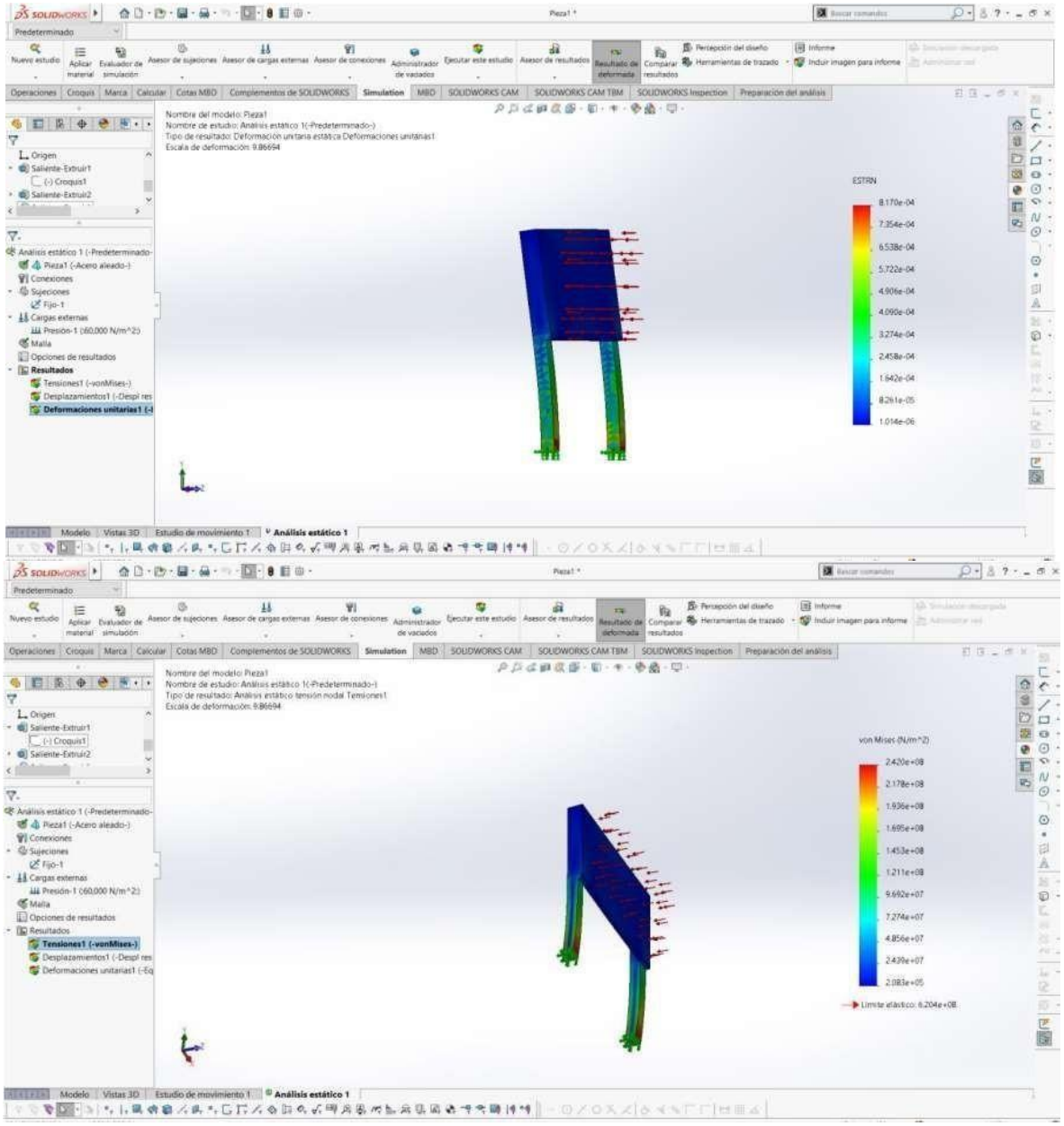
Figura 2. Espacio de diseño

Se tomarán ciertas consideraciones para la solución de esta práctica: 5 cargas, los apoyos tendrán restricciones en “X”,” Y” y el espacio de diseño para esta práctica será de:



Implementación o desarrollo del prototipo

Análisis de esfuerzos en la estructura con una carga uniforme en SolidWorks.



Procedimiento de la programación

Usaremos el código de la práctica 1 dándole unos cambios necesarios para poder implementar de manera exitosa la nueva implementación, la cual en este caso es el panorámico.

Código original

```
function toppract(nelx,nely,volfrac,penal,rmin)
% 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;
dc(ely,elx) = -penal*x(ely,elx)^(penal-1)*Ue'*KE*Ue;
end
end
% FILTERING OF SENSITIVITIES
[dc] = check(nelx,nely,rmin,x,dc);
% DESIGN UPDATE BY THE OPTIMALITY CRITERIA METHOD
[x] = OC(nelx,nely,x,volfrac,dc);
% PRINT RESULTS
change = max(max(abs(x-xold)));
disp(['It.:' sprintf('%4i',loop) 'Obj.:' sprintf('%10.4f',c) ...
'Vol.:' sprintf('%6.3f',sum(sum(x))/(nelx*nely)) ...
'ch.:' sprintf('%6.3f',change )])
% PLOT DENSITIES
colormap(gray); imagesc(-x); axis equal; axis tight; axis off; pause(1e-6);
end

%%%%%%%%%% OPTIMALITY CRITERIA UPDATE %%%%%%%%%%
function [xnew]=OC(nelx,nely,x,volfrac,dc)
l1 = 0;
l2 = 100000;
move = 0.2;
while (l2-l1 > 1e-4)
lmid = 0.5*(l2+l1);
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);
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 = union([1:2:2*(nely+1)],[2*(nelx+1)*(nely+1)]);
alldofs = [1:2*(nely+1)*(nelx+1)];
freedofs = setdiff(alldofs,fixeddofs);
% SOLVING
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)];
```

Fig.1 código original

Cambios para fuerzas múltiples

Se tiene que editar el script para poder ingresar las fuerzas necesarias, observando vemos que tenemos 5 y para cambiar el anclaje del espacio de diseño a otra posición se tiene que cambiar con la instrucción *fixeddofs* modificando ciertas líneas

```
71      %%%%%%%%%% FE-ANALYSIS %%%%%%%%%%%%
72      function [U]=FE(nelx,nely,x,penal)
73 -      [KE] = lk;
74 -      K = sparse(2*(nelx+1)*(nely+1),2*(nelx+1)*(nely+1));
75 -      F = sparse(2*(nely+1)*(nelx+1),1);
```

FIg.2 Fragmento del código original

```
86      %%%%%%%%%% FE-ANALYSIS %%%%%%%%%%%%
87      function [U]=FE(nelx,nely,x,penal)
88 -      [KE] = lk;
89 -      K = sparse(2*(nelx+1)*(nely+1),2*(nelx+1)*(nely+1));
90 -      F = sparse(2*(nely+1)*(nelx+1),5);
91 -      U = sparse(2*(nely+1)*(nelx+1),5);
```

Fig.3 Fragmento del código con líneas modificadas

```
17 -  for ely = 1:nely
18 -  for elx = 1:nelx
19 -    n1 = (nely+1)*(elx-1)+ely;
20 -    n2 = (nely+1)* elx +ely;
21 -    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);
22 -    c = c + x(ely,elx)^penal*Ue'*KE*Ue;
23 -    dc(ely,elx) = -penal*x(ely,elx)^(penal-1)*Ue'*KE*Ue;
24 -  end
25 -  end
```

Fig.4 Fragmento del código original

```
28 -  for ely = 1:nely
29 -  for elx = 1:nelx
30 -    n1 = (nely+1)*(elx-1)+ely;
31 -    n2 = (nely+1)* elx +ely;
32 -    dc(ely,elx) = 0.;
33 -  for i = 1:5
34 -    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);
35 -    c = c + x(ely,elx)^penal*Ue'*KE*Ue;
36 -    dc(ely,elx) = -penal*x(ely,elx)^(penal-1)*Ue'*KE*Ue;
37 -  end
38 -  end
39 -  end
```

Fig.5 Fragmento del código con líneas modificadas

```
85      % DEFINE LOADSAND SUPPORTS(HALF MBB-BEAM)
86 -    F(2,1) = -1;
87 -    fixeddofs = union([1:2:2*(nely+1)],[2*(nelx+1)*(nely+1)]);
88 -    alldofs = [1:2*(nely+1)*(nelx+1)];
```

Fig.6 Fragmento del código original

```
100     % DEFINE LOADSAND SUPPORTS(HALF MBB-BEAM)
101 -   F(2*(nelx)*(nely+1)+2,1) = 1;
102 -   F(2*(nelx)*(nely+1)+(nely/4),2) = 1;
103 -   F(2*(nelx)*(nely+1)+(nely/2),3) = 1;
104 -   F(2*(nelx)*(nely+1)+(nely),4) = 1;
105 -   F(2*(nelx)*(nely+1)+(nely*1.2),5) = 1;
106 -   fixeddofs = 2*(nely+1):2*(nely+1):2*(nelx+1)*(nely+1);
107 -   alldofs = [1:2*(nely+1)*(nelx+1)];
108 -   freedofs = setdiff(alldofs,fixeddofs);
```

Fig.7 Fragmento del código con líneas modificadas

Empotramiento diagonal

Para nosotros poder crear el empotramiento diagonal o también conocido como espacio en blanco en la parte inferior derecha, necesitamos modificar el codigo original para poder recrear el espacio conocido con los siguientes cambios en la codificación para llegar al resultado del panorámico.

```
3    function toppract(nelx,nely,volfrac,penal,rmin)
4      % INITIALIZE
5 -    x(1:nely,1:nelx) = volfrac;
6 -    loop = 0;
7 -    change = 1.;
```

Fig.8 Fragmento del código original

```
3    function toppract(nelx,nely,volfrac,penal,rmin)
4    % INITIALIZE
5 -  x(1:nely,1:nelx) = volfrac;
6 -  loop = 0;
7    %DECLARACION DE VACIO
8 -  for ely = 1:nely
9 -      for elx = 1:nelx
10 -         if (((ely-(nely*0.5)<(2*elx)-(1.36*nelx)) |(ely<(1+nely*0.5)))&(elx >(1+nelx)*0.6666))
11 -             passive(ely,elx)=1;
12 -         else
13 -             passive(ely,elx) = 0;
14 -         end
15 -     end
16 - end
17 - x(find(passive))=0.001;
18 - change = 1.;
```

Fig.9 Fragmento del código con líneas modificadas

```
28     % DESIGN UPDATE BY THE OPTIMALITY CRITERIA METHOD
29 -   [x] = OC(nelx,nely,x,volfrac,dc);
```

Fig.10 Fragmento del código original

```
42   % DESIGN UPDATE BY THE OPTIMALITY CRITERIA METHOD
43 - [x] = OC(nelx,nely,x,volfrac,dc,passive);
```

Fig.11 Fragmento del código con líneas modificadas

```
39     %%%%%%%%%% OPTIMALITY CRITERIA UPDATE %%%%%%%%%%
40   function [xnew]=OC(nelx,nely,x,volfrac,dc)
```

Fig.12 Fragmento del código original

```
53     %%%%%%%%%% OPTIMALITY CRITERIA UPDATE %%%%%%%%%%
54   function [xnew]=OC(nelx,nely,x,volfrac,dc,passive)
```

Fig.13 Fragmento del código con líneas modificadas

```
44 - while (l2-l1 > 1e-4)
45 -   lmid = 0.5*(l2+l1);
46 -   xnew = max(0.001,max(x-move,min(1.,min(x+move,x.*sqrt(-dc./lmid)))));
47 -   if sum(sum(xnew)) - volfrac*nelx*nely > 0;
```

Fig.14 Fragmento del código original

```
60 -   xnew = max(0.001,max(x-move,min(1.,min(x+move,x.*sqrt(-dc./lmid)))));
61 -   xnew(find(passive)) = 0.001;
62 -   if sum(sum(xnew)) - volfrac*nelx*nely > 0;
```

Fig.15 Fragmento del código con líneas modificadas

Código final

```
%Práctica #3 Laboratorio Biomecánica Equipo 7

function toppract(nelx,nely,volfrac,penal,rmin)
% INITIALIZE
x(1:nely,1:nelx) = volfrac;
loop = 0;
%DECLARACION DE VACIO
for ely = 1:nely
    for elx = 1:nelx
        if (((ely-(nely*0.5)<(2*elx)-(1.36*nelx)) |(ely<(1+nely*0.5)))&(elx >(1+nelx)*0.6666))
            passive(ely,elx)=1;
        else
            passive(ely,elx) = 0;
        end
    end
end
x(find(passive))=0.001;
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;
dc(ely,elx) = 0.;
for i = 1:5
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;
dc(ely,elx) = -penal*x(ely,elx)^(penal-1)*Ue'*KE*Ue;
end
end
end
% FILTERING OF SENSITIVITIES
[dc] = check(nelx,nely,rmin,x,dc);
% DESIGN UPDATE BY THE OPTIMALITY CRITERIA METHOD
[x] = OC(nelx,nely,x,volfrac,dc,passive);
% PRINT RESULTS
change = max(max(abs(x-xold)));
disp(['It.:' sprintf('%4i',loop) 'Obj.:' sprintf('%10.4f',c) ...
'Vol.:' sprintf('%6.3f',sum(sum(x))/(nelx*nely)) ...
'ch.:' sprintf('%6.3f',change )])
% PLOT DENSITIES
colormap(gray); imagesc(-x); axis equal; axis tight; axis off; pause(1e-6);
end
```

```
52
53      %%%%%%%%%% OPTIMALITY CRITERIA UPDATE %%%%%%%%%
54      function [xnew]=OC(nelx,nely,x,volfrac,dc,passive)
55 -    l1 = 0;
56 -    l2 = 100000;
57 -    move = 0.2;
58 -    while (l2-l1 > 1e-4)
59 -    lmid = 0.5*(l2+l1);
60 -    xnew = max(0.001,max(x-move,min(1.,min(x+move,x.*sqrt(-dc./lmid)))));
61 -    xnew(find(passive)) = 0.001;
62 -    if sum(sum(xnew)) - volfrac*nelx*nely > 0;
63 -    l1 = lmid;
64 -    else
65 -    l2 = lmid;
66 -    end
67 -    end
68
69      %%%%%%%%%% MESH-INDEPENDENCY FILTER %%%%%%%%%%%%
70      function [dcn]=check(nelx,nely,rmin,x,dc)
71 -    dcn=zeros(nely,nelx);
72 -    for i = 1:nelx
73 -    for j = 1:nely
74 -    sum=0.0;
75 -    for k = max(i-round(rmin),1):min(i+round(rmin),nelx)
76 -    for l = max(j-round(rmin),1):min(j+round(rmin), nely)
77 -    fac = rmin-sqrt((i-k)^2+(j-l)^2);
78 -    sum = sum+max(0,fac);
79 -    dcn(j,i) = dcn(j,i) + max(0,fac)*x(l,k)*dc(l,k);
80 -    end
81 -    end
82 -    dcn(j,i) = dcn(j,i)/(x(j,i)*sum);
83 -    end
84 -    end
85
86      %%%%%%%%%% FE-ANALYSIS %%%%%%%%%%%%%
87      function [U]=FE(nelx,nely,x,penal)
88 -    [KE] = lk;
89 -    K = sparse(2*(nelx+1)*(nely+1),2*(nelx+1)*(nely+1));
90 -    F = sparse(2*(nely+1)*(nelx+1),5);
91 -    U = sparse(2*(nely+1)*(nelx+1),5);
92 -    for ely = 1:nely
93 -    for elx = 1:nelx
94 -    n1 = (nely+1)*(elx-1)+ely;
95 -    n2 = (nely+1)*elx+ely;
96 -    edof = [2*n1-1; 2*n1; 2*n2-1; 2*n2; 2*n2+1; 2*n2+2; 2*n1+1; 2*n1+2];
97 -    K(edof,edof) = K(edof,edof)+x(ely,elx)^penal*KE;
98 -    end
99 -    end
100     % DEFINE LOADSAND SUPPORTS(HALF MBB-BEAM)
101 -   F(2*(nelx)*(nely+1)+2,1) = 1;
102 -   F(2*(nelx)*(nely+1)+(nely/4),2) = 1;
103 -   F(2*(nelx)*(nely+1)+(nely/2),3) = 1;
```

```
104 -    F(2*(nelx)*(nely+1)+(nely),4) = 1;
105 -    F(2*(nelx)*(nely+1)+(nely*1.2),5) = 1;
106 -    fixeddofs = 2*(nely+1):2*(nely+1):2*(nelx+1)*(nely+1);
107 -    alldofs = [1:2*(nely+1)*(nelx+1)];
108 -    freedofs = setdiff(alldofs,fixeddofs);
109      % SOLVING
110 -    U(freedofs,:) = K(freedofs,freedofs) \ F(freedofs,:);
111 -    U(fixeddofs,:)= 0;
112
113      %%%%%%%%%% ELEMENT STIFFNESS MATRIX %%%%%%%
114      function [KE]=lk
115 -    E = 1.;
116 -    nu = 0.3;
117 -    k=[ 1/2-nu/6 1/8+nu/8 -1/4-nu/12 -1/8+3*nu/8 ...
118      -1/4+nu/12 -1/8-nu/8 nu/6 1/8-3*nu/8];
119 -    KE = E/(1-nu^2)*[ k(1) k(2) k(3) k(4) k(5) k(6) k(7) k(8)
120      k(2) k(1) k(8) k(7) k(6) k(5) k(4) k(3)
121      k(3) k(8) k(1) k(6) k(7) k(4) k(5) k(2)
122      k(4) k(7) k(6) k(1) k(8) k(3) k(2) k(5)
123      k(5) k(6) k(7) k(8) k(1) k(2) k(3) k(4)
124      k(6) k(5) k(4) k(3) k(2) k(1) k(8) k(7)
125      k(7) k(4) k(5) k(2) k(3) k(8) k(1) k(6)
126      k(8) k(3) k(2) k(5) k(4) k(7) k(6) k(1)];
```

Después de tener los cambios necesarios en el código se optó por darle un valor a toppract de toppract(40,80,0.2,3.0,0.5) en la ventana de comando de Matlab. Llegando a nuestra respuesta final lacual es esta:

```
Command Window
>>
>> toppract(40,80,0.2,3.0,0.5)
It.:    1Obj.:42819847281.0592Vol.: 0.200ch.: 0.200
Warning: MATLAB has disabled some advanced graphics rendering features by switching to software OpenGL. For more
information, click here.
It.:    2Obj.:42819839234.7122Vol.: 0.200ch.: 0.200
It.:    3Obj.:42819838251.6167Vol.: 0.200ch.: 0.200
It.:    4Obj.:42819837687.0867Vol.: 0.200ch.: 0.200
It.:    5Obj.:42819837275.7362Vol.: 0.200ch.: 0.200
It.:    6Obj.:42819837054.2473Vol.: 0.200ch.: 0.200
It.:    7Obj.:42819836942.4612Vol.: 0.200ch.: 0.200
```

Fig.16 Ventana de comando con el inicio de implementación del diseño con los parámetros dados [toppract(40,80,0.2,3.0,0.5)]

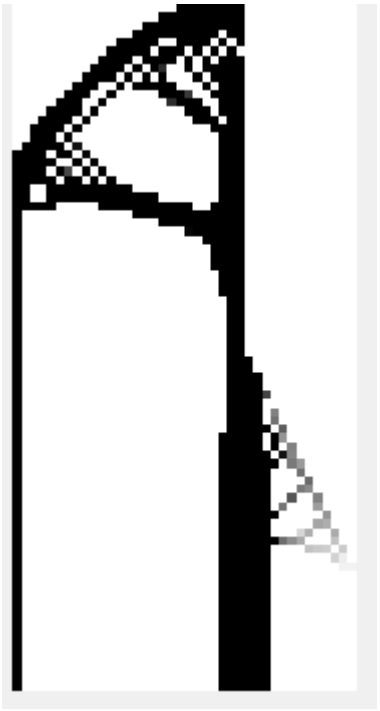


Fig.17 Imagen del diseño resultante con los parámetros dados en la ventana de comando en Matlab.

Conclusiones

Andrés Anaya Hernández

En esta actividad se observó el comportamiento estático de un panorámico y al cambiar ciertas partesdel código, esta muestra ahora una representación del ya mencionado panorámico, con esto se analizamás a fondo dicho código y como con solo cambiar algunos valores o datos puede crear algo nuevo.

Carlos Eduardo Rivera López

En conclusión, en esta actividad pude volver a apreciar la versatilidad de utilizar este código, iniciamosviendo una viga, después el marco de una bicicleta y ahora vemos la estructura de un panorámico, el poder hacer simulaciones nos permite evaluar los materiales y formas que utilizaríamos físicamente, pero con la ventaja de que no gastamos tantos recursos para ello.

César Armando Luna

Con esta práctica conocí las fuerzas y resistencias que tienen los panorámicos, desde que se piensa en donde se colocará, los materiales de la estructura, entre otras cosas que intervienen; en el softwarede Matlab se realizó la simulación del desempeño mecánico de componentes en el cual observamos el análisis por medio de elemento finito.

Víctor Adrián Higuera Vázquez

Gracias a esta práctica se aprendió a analizar de una manera diferente el comportamiento estático deun panorámico y así poder plasmarlo en un código en Matlab y llevarlo a su representación gráfica conel mismo. Esta práctica resulto de alto valor de aprendizaje para mí, al ponernos como reto el analizarde manera critica el programa original y con eso poder llegar a la representación gráfica del panorámico.

Gabryiel Bailon Avila

Con esta actividad pudimos implementar la codificación para aplicar la optimización topológica a un diseño de un panorámico, por lo que considero que el código tiene mucha versatilidad para ser usado en otros diseños estructurales, claro siempre tomando en cuenta las restricciones tanto 2D como 3D. Como resultado de la optimización, se obtuvo una imagen de la reducción de la fracción volumétrica del espacio de diseño, conforme a lo calculado por la codificación y lo esperado analíticamente hablando.

Referencias Bibliográficas

- 99 Line Topology Optimization Code – O. Sigmund, Department of Solid Mechanics, Building404, Technical University of Denmark, DK-2800 Lyngby, Denmark.
- Blu Cactus. (2021). Estructura de un panorámico. octubre 2021, de Blu Cactus Digital MarketingSitio web: https://www.blucactus.com.mx/cuanto-cuesta-estructura-panoramico/