UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN

FACULTAD DE INGENIERÍA MECÁNICA Y ELÉCTRICA

LABORATORIO DE BIOMECÁNICA

PRÁCTICA 3: Diseño de la estructura de un panorámico.

ING. YADIRA MORENO VERA

ING. ISAAC ESTRADA

Luis Alejandro Salais Meza	José Juan García Martínez	Daniel García Rodarte	Raymundo López Mata	Gerardo Antonio Contreras Sandate
1877483	1911641	1912044	1923217	1860063
IMTC	IMTC	IMTC	IMTC	IMTC

BRIGADA: 509 **AGOSTO – DICIEMBRE 2022**

SALÓN 12BMC **VIERNES N5**

Ciudad Universitaria, San Nicolás de los Garza, Nuevo León a 21 de octubre de 2022

Diseño de la Estructura de un Panorámico

Introducción

En el mundo actual, conocemos diversas metodologías para la solución de problemas, así como herramientas para realizar una mejor convención de estar en lo correcto, así también en el mundo de la simulación y diseño se necesitan más de un software o programas para dar a conocer una respuesta correcta. En está practica se llevará a cabo el deseño de la estructura de un panorámico con la ayuda de la programación de 99 líneas para desarrollar un buen diseño con programación en Matlab.

Objetivo

Desarrollar en el estudiante la capacidad de análisis, implementación y solución de un problema propuesto.

Estado de Arte

Los panorámicos se exponen a altas ráfagas de viento, por lo que su estructura ocupa ser muy rígida para soportar estas fuerzas. El espacio de diseño a evaluar será de 2 dimensiones, las cargas y los apoyos de observan en la figura 3.1

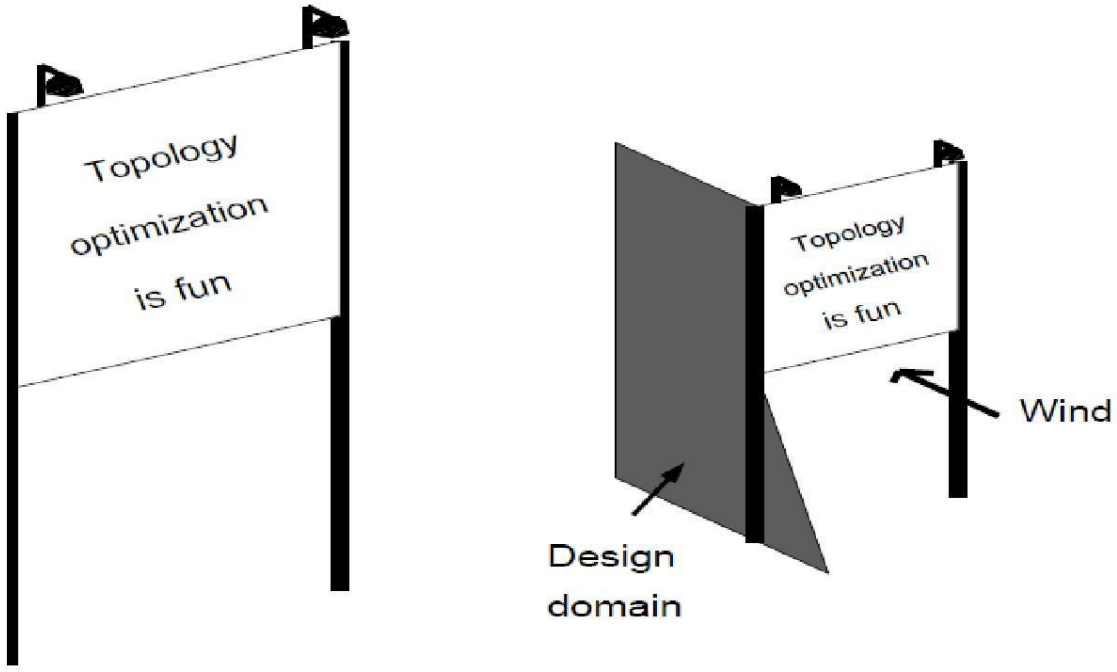


Figura 3.1 Imagen del Panorámico

Diseño de la Geometría, Alcances y Limitaciones

En la figura 3.2 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 las que tendrán el propósito de diseño de poder soportar cualquier viento en contra y así se genera un buen diseño.

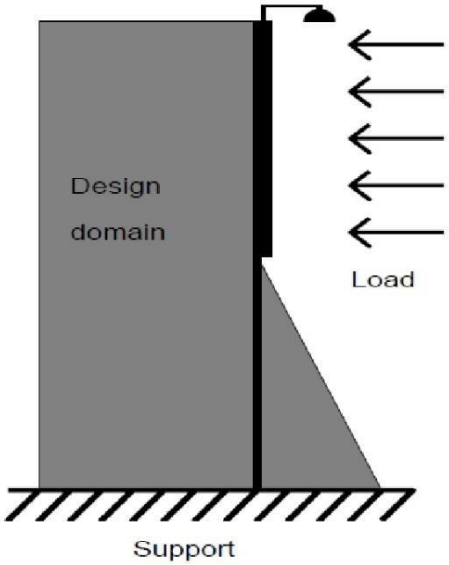
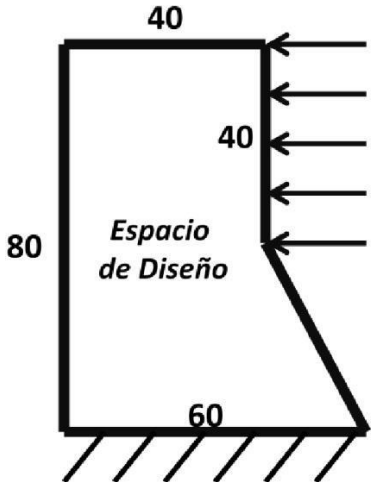


Figura 3.2 Espacio de diseño

Se tomarán ciertas consideraciones para la solución de esta práctica, se simulará teniendo 5 cargas los cuales representan los vientos que chocarán con nuestro panorámico para dar a conocer si nuestro diseño está bien definido, también los apoyos tendrán restricciones en “X”,”Y” y el espacio de diseño para esta práctica será de:



Pasos del Desarrollo de la Programación

- Realizar el ejercicio con el espacio de diseño propuesto

Fuerzas múltiples

Para empezar, tenemos que editar nuestro script topp, se tiene guardado como topp3, para poder ingresar las fuerzas que requerimos, si observamos nos encontramos con 5 y para cambiar el anclaje del espacio de diseño a otra posición se tiene que cambiar la línea con la instrucción *fixeddofs*, para esto se modificaran las siguientes líneas:

```
%%%%%%%%%% 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 = zeros(2*(nely+1)*(nelx+1),1);
```

Figura 1. Código original

```
%%%%%%%%%% 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),5);
U = sparse(2*(nely+1)*(nelx+1),5);
```

Figura 2. Líneas modificadas

```
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
```

Figura 3. Código original

```
    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],i);
    c = c + x(ely,elx)^penal*Ue'*KE*Ue;
    dc(ely,elx) = dc(ely,elx) - penal*x(ely,elx)^(penal-1)*Ue'*KE*Ue;
  end
```

Figura 4. Líneas modificadas

```
% DEFINE LOADS AND 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)];
```

Figura 5. Código original

```
% DEFINE LOADS AND SUPPORTS (HALF MBB-BEAM)
F(2*(nelx)*(nely+1)+2,1)=1;
F(2*(nelx)*(nely+1)+20,1)=1;
F(2*(nelx)*(nely+1)+40,1)=1;
F(2*(nelx)*(nely+1)+60,1)=1;
F(2*(nelx)*(nely+1)+80,1)=1;
fixeddofs   = 2*(nely+1):2*(nely+1):2*(nelx+1)*(nely+1);
```

Figura 6. Líneas modificadas

Empotramiento diagonal (elementos pasivos)

Para crear el empotramiento diagonal, o crear el espacio en blanco para recrear el empotramiento en la parte inferior derecha; en el archivo del uso del código de 99 líneas existe una sección donde se habla de elementos pasivos el cual sirve de ayuda para determinar un espacio en blanco, en el ejemplo del archivo viene como hacer un círculo, y nosotros necesitamos un rectángulo y un triángulo para esto se modificaron y/o agregaron las siguientes líneas:

```
%%%% A 99 LINE TOPOLOGY OPTIMIZATION CODE BY OLE SIGMUND, JANUARY 2000 %%%
function top(nelx,nely,volfrac,penal,rmin);
% INITIALIZE
x(1:nely,1:nelx) = volfrac;
loop = 0;
change = 1.;
% START ITERATION
```

Figura 7. Código original

```
for ely = 1:nely
    for elx = 41:nelx
        if elx - 20 < (ely/2)
            passive(ely,elx)=0;
        else
            passive(ely,elx)=1;
        end
    end
end
x(find(passive))= 0.001;
```

Figura 8. Líneas modificadas

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

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

Figura 9. Código original

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

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

Figura 10. Líneas modificadas

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

Figura 11. Código original

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

Figura 12. Líneas modificadas

Resultado de la optimización

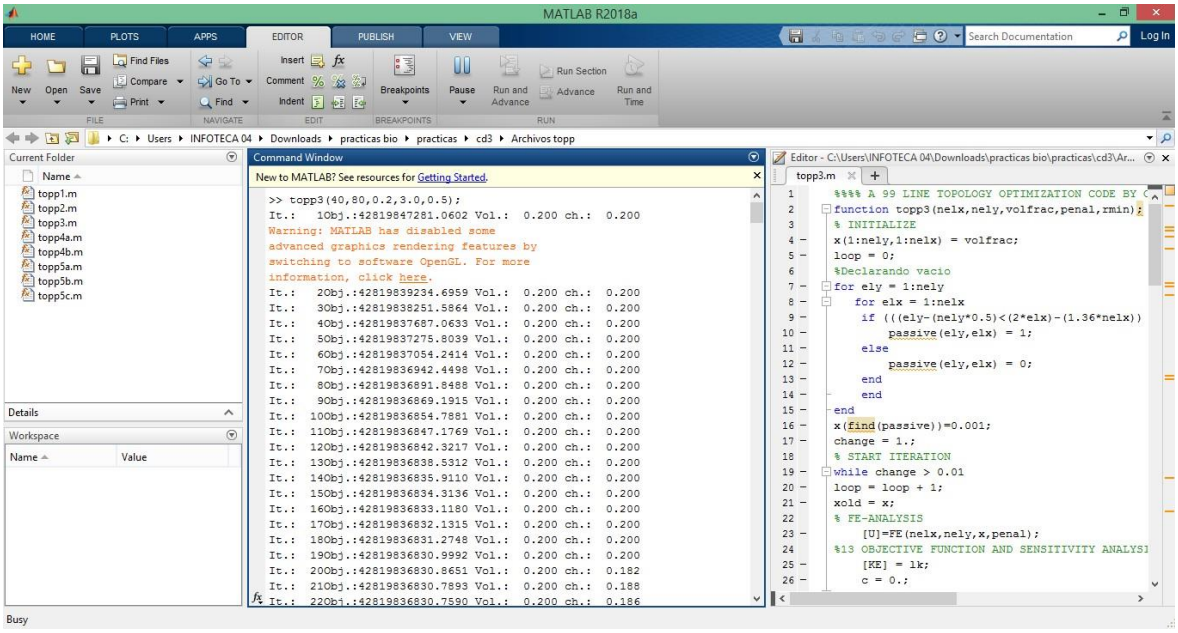


Figura 13. Resultado en la ventana de comando y editor

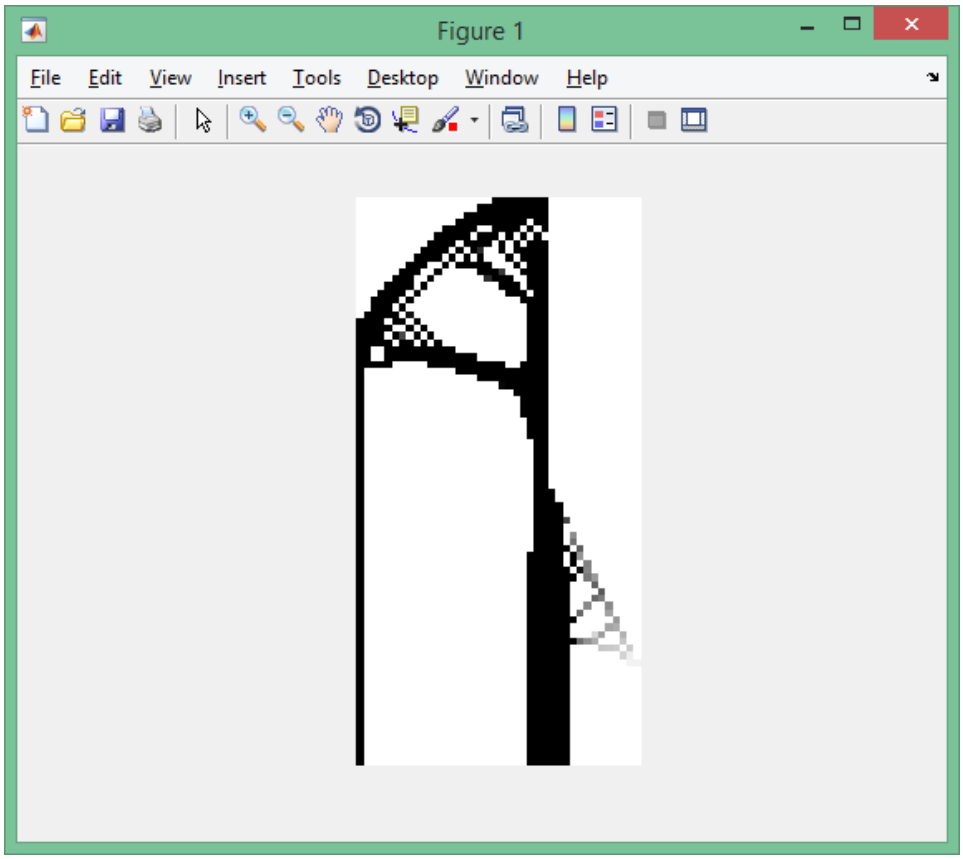


Figura 14. Diseño resultante para el problema planteado

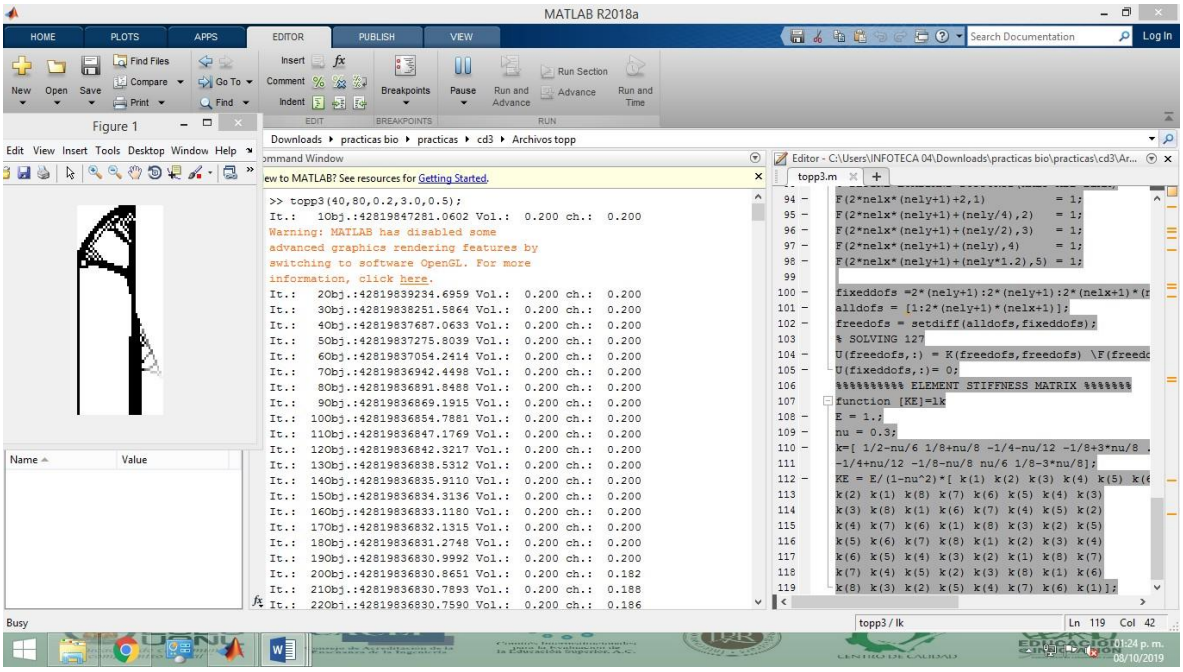
Código completo con las modificaciones finales

```
%%%% A 99 LINE TOPOLOGY OPTIMIZATION CODE BY OLESIGMUND,
OCTOBER 1999 %%%
function topp3(nelx,nely,volfrac,penal,rmin);
% INITIALIZE
x(1:nely,1:nelx) = volfrac; loop =
0;
%Declarando 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);
 %13 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; %19
            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) = dc(ely,elx)-penal*x(ely,elx)^(penal-1)*Ue'*KE*Ue;
            end
      end
end
%25 FILTERING OF SENSITIVITIES
[dc] = check(nelx,nely,rmin,x,dc);
%27 DESIGN UPDATE BY THE OPTIMALITY CRITERIA METHOD
[x] = OC(nelx,nely,x,volfrac,dc,passive);
```

```
%29 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
%40 %%%%%%%%%% OPTIMALITY CRITERIA UPDATE %%%%%%%%%%
function [xnew]=OC(nelx,nely,x,volfrac,dc,passive)
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)))));
xnew(find(passive)) = 0.001;
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
%65 %%%%%%%%%% 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),5); U =zeros(2*(nely+1)*(nelx+1),5); 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*nelx*(nely+1)+2,1)                = 1;
F(2*nelx*(nely+1)+(nely/4),2)   = 1;
F(2*nelx*(nely+1)+(nely/2),3)   = 1;
F(2*nelx*(nely+1)+(nely),4)          = 1;
F(2*nelx*(nely+1)+(nely*1.2),5) = 1;

fixeddofs =2*(nely+1):2*(nely+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)];
```

Programa en Matlab



Conclusiones

José Juan García Martínez

En está practica se realizó el diseño de un panorámico en el cual se tomaron algunas medidas como las afectaciones por los vientos, los cuales en la vida actual tienden a ser muy imprevistos y desconocidos al momento de todo los días, ya que esto causa que se debiliten o puedan reducir su vida útil. También, el diseño y simulación se realizó en Matlab con la programación de 99 líneas el cuál nos facilitó gracias a su optimización ya que es un sistema largo.

Daniel García Rodarte

En conclusión, durante el desarrollo de la práctica se utilizó el software de Matlab para el diseño de un panorámico, gracias a este tipo de software se pueden crear diseños y simulaciones en donde se toman en cuenta medidas reales para analizar su comportamiento aproximado al momento de ejecutarlos en alguno prototipo. Gracias a Matlab podemos realizar el diseño y simulación de un panorámico optimizando en cada momento el diseño propuesto desde el principio.

Gerardo Antonio Contreras Sandate

En esta práctica donde se simuló el diseño de la estructura de un panorámico, se utilizó el software de Matlab para crear el diseño y simulación de dicha estructura, con esto podemos experimentar la capacidad de análisis, implementación y solución de un problema solamente con simulaciones, para que el costo de estas pruebas sea menor al momento de que suceda en la industria. En cuanto al problema propuesto en esta práctica, es importante saber como lo puede afectar el viento, que es algo que va a estar en constante contacto con el diseño.

Raymundo López Mata 1923217

Con la realización de esta tercer práctica pude entender un poco la programación realizada en Matlab, ya que es la primer práctica que se realiza utilizando este software. Me pareció interesante ver que con unas modificaciones al código se pueden realizar análisis estructurales como los esfuerzos, así como optimizando la estructura y los esfuerzos aplicados en ésta debido a las ráfagas de viento que influyen en este caso en un panorámico.

Luis Alejandro Salais Meza

Con la realización de esta práctica se pudo analizar una situación compleja como lo son las fuerzas que soporta un panorámico y desarrollar una simulación de la misma tomando en cuenta dichas fuerzas, así como los soportes de empotramiento que estos panorámicos tienen para producir resultados que permitan un análisis posterior. Este análisis se realizo por medio de Matlab, haciendo uso de una programación especifica para este tipo de casos, haciendo algunas modificaciones al mismo para adecuarse perfectamente a la situación planteada.