Parte 1. Traslación y rotación con elementos matriciales.

Considere como punto de referencia de información su fecha de cumpleaños, asuma de manera genérica que es: DD/MM/AAAA.

DD/MM/AAAA = 04/07/1997

Considere el punto P cuyas coordenadas son: P = (DD , MM) [Metros].
P = (04 , 07) [Metros].
Considere un ángulo alfa, cuyo valor es: alfa = DD + MM + 10 [Grados].
alfa = 04 + 07 + 10 = 21 [Grados]
Si el año de su nacimiento, se expresa como: A_4 A_3 A_2 A_1
A_4 A_3 A_2 A_1 = 1,9,9,7
Considere la distancia d, calculada como: d = A_4 + A_3 + A_2 + A_1 [Metros]
d = 1 + 9 + 9 +7 = 26 [Metros]

Para que funcione nuestro programa se utilizarán los archivos auxiliares provistos en el curso:

Traslacion2D.m

```
# Recibe como entrada un Punto del plano
# P=[x y] y un vector v=[vx vy]
# Retorna las coordenadas del punto resultante de la traslacion.

function Punto_T = Traslacion2D( Punto, Vector )

  Punto_T(1) = Punto(1) + Vector(1);
  Punto_T(2) = Punto(2) + Vector(2);

endfunction
```

Rotacion2D.m

```
# Recibe como entrada un Punto del plano P=[x y], un Centro y un Angulo [radianes] realiza la rotación tomando el origen como centro de giro
# Retorna las coordenadas del punto resultante de la traslacion.

function Punto_R = Rotacion2D(Punto, Centro, Angulo)
  % Arma la matriz de rotacion
 M = [cos(Angulo) -sin(Angulo); sin(Angulo) cos(Angulo)]

 %Mueve el punto para lograr equivalencia de giro en el origen
  Pt_aux = Traslacion2D(Punto, -Centro);

  % Procesa la rotacion en el origen
  Pr_aux = M * [Pt_aux(1); Pt_aux(2)];

  % Traslada el resultado, restituyendo posicion
  Punto_R = Traslacion2D(Pr_aux, Centro);

endfunction
```

Traslacion2DTrack.m

```
# Realiza un trazado de la trayectoria, a partir del punto de partida (Punto)
# el vector de traslacion (Vector) y la cantidad de partes a segmentar (Nparte).
# Retorna la cantidad de elementos

function [ vec_track_x, vec_track_y ] = Traslacion2DTrack(Punto, Vector, Nparte)
  pos = 1;
  % Inicia vectores de coordenadas de puntos de trayecto
  vec_track_x(pos) = Punto(1);
  vec_track_y(pos) = Punto(2);
  pos = pos + 1;

  P_tra = Traslacion2D( Punto, Vector );

  %-------------------------------------------------------------------
  Vp(1) = Vector(1)/Nparte;
  Vp(2) = Vector(2)/Nparte;

  Vi = Vp;

  while ( abs(Vi(1)) < abs(Vector(1)) )

    % Trasladar el punto P segun Vector
    Pi = Traslacion2D( Punto, Vi);
    Vi = Vi + Vp;

    % Actualiza vectores de trayecto
    vec_track_x(pos) = Pi(1);
    vec_track_y(pos) = Pi(2);
    pos = pos + 1;

  endwhile

  % Agrega el punto final del tramo
    vec_track_x(pos) = P_tra(1);
    vec_track_y(pos) = P_tra(2);

endfunction
```

Rotacion2DTrack.m

```
# Realiza un trazado de la trayectoria, a partir del punto de partida
# el vector de traslacion y la cantidad de partes a segmentar.
# Retorna la cantidad de elementos

function [ vec_track_x, vec_track_y ] = Rotacion2DTrack( Punto, Centro, Angulo_rad, Nparte )

 pos = 1;
 Ap = Angulo_rad / Nparte;        % Segmento de ángulo

 % Rotar el punto Px segunto Angulo_rad, alrededor del origen
 Px = Punto;
 Ai = Ap;

 P_rot = Rotacion2D( Punto, Centro, Angulo_rad );

 while (abs(Ai) < abs(Angulo_rad))

  Pi = Rotacion2D( Px, Centro, Ai );
  Ai = Ai + Ap;

  % Actualiza vectores de trayecto
  vec_track_x(pos) = Pi(1);
  vec_track_y(pos) = Pi(2);
  pos = pos + 1;

 endwhile

 % Agrega el punto final del tramo
  vec_track_x(pos) = P_rot(1);
  vec_track_y(pos) = P_rot(2);

endfunction
```

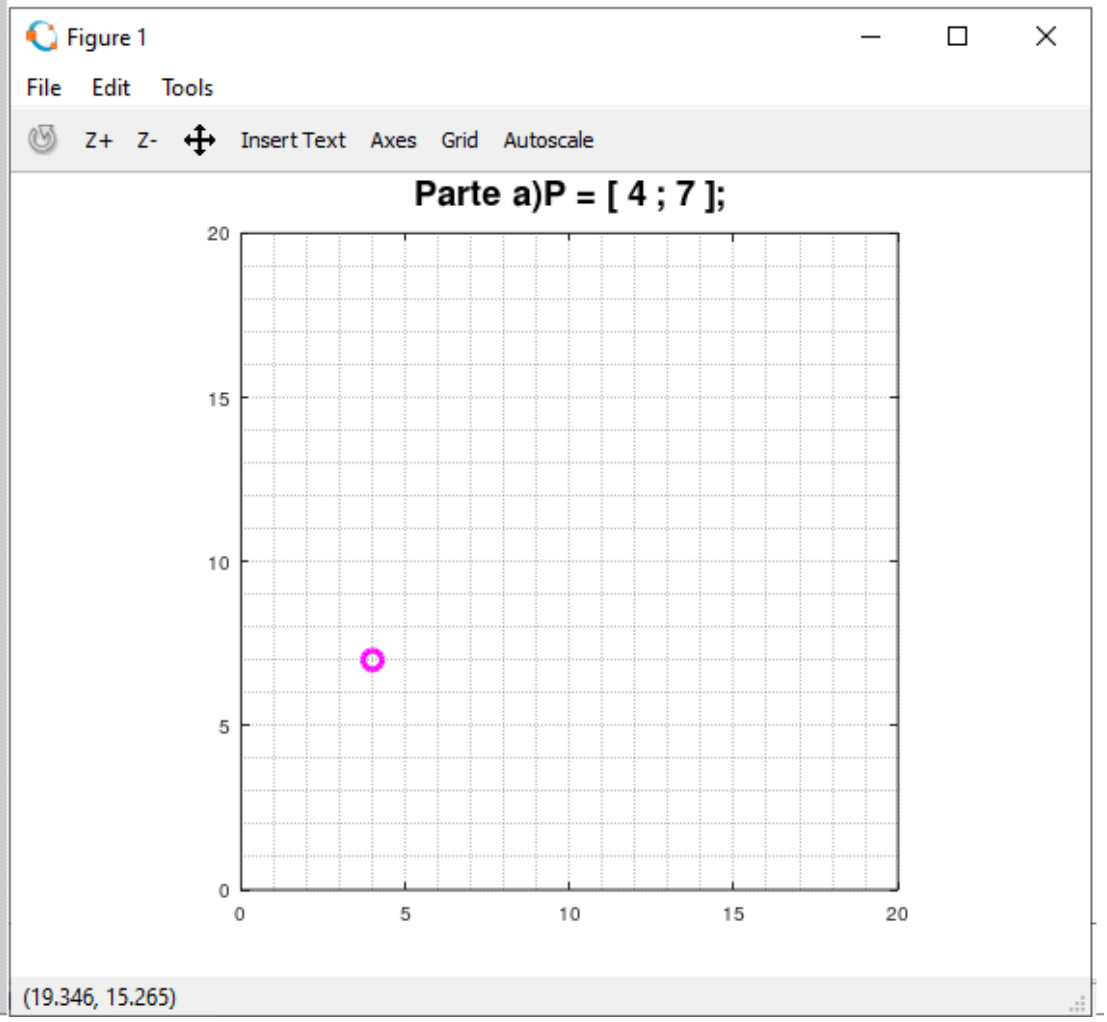
Nota: para implementar la solución nos basaremos en el archivo **Movimientos_Ej2_1.m**.

a) Represente el punto P en el sistema de coordenadas

Para empezar crearemos el archivo **parte1.m**, donde principalmente definiremos la matriz **P = [4 ; 7]** (día y mes de nacimiento) y realizaremos el ploteado de esos datos sobre una gráfica, en este caso **P(1)** corresponde al **4** y **P(2)** al **7**.

*parte1.m

```
1  %  Proyecto: Parte 1. Traslación y rotación con elementos matriciales
2  %  Autor: Andres Romano
3  %  Fecha: 30 de setiembre de 2021
4  %  Ultima modificaciÃ³n: 4 de octubre de 2021
5  clear
6  clc
7
8  P = [ 4 ; 7 ];
9
10 %--------------------------------------------------------------------------
11 figure()
12 plot(P(1),P(2), 'mo-', 'markersize', 10,'linewidth', 3); grid minor;
13 title("Parte a)P = [ 4 ; 7 ];", "fontSize", 18);
14 axis([0,20, 0,20])
15 axis("square")
```



b) Realice una rotación del punto P, una cantidad angular igual a alfa alrededor del origen (0,0).

- Ahora necesitamos definir matrices para el **origen**, el **ángulo de rotación** y el **centro de rotación** que será igual al origen.
- Después implementaremos una línea de código para convertir Grados a Radianes.
- Como tercer punto guardaremos en una variable la salida de la llamada a **Rotacion2D()** con los parámetros correspondientes.
- Por último graficamos los datos de **P_rot.**

```
%  Proyecto: Parte 1. Traslación y rotación con elementos matriciales
%  Autor: Andres Romano
%  Fecha: 30 de setiembre de 2021
%  Ultima modificaciÃ³n: 4 de octubre de 2021
clear
clc
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% PARAMETROS:

Origen     = [ 0 ; 0];
Alfa_gra = 21;              % Angulo de rotacion en grados
Centro_rot = Origen;          % Punto centro de rotacion

P = [ 4 ; 7 ];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Alfa_rad = Alfa_gra * (pi / 180); % Convierte grados en radianes
%-------------------------------------------------------------------------------
% Calculo directo de posiciones finales de los movimientos

P_rot = Rotacion2D( P, Centro_rot, Alfa_rad );
%-------------------------------------------------------------------------------
figure()
plot(P(1),P(2), 'mo-', 'markersize', 10,'linewidth', 3); grid minor;
title("Parte a)P = [ 4 ; 7 ];", "fontSize", 18);
axis([0,20, 0,20])
axis("square")

axis([0,20, 0,20])
axis("square")
%-------------------------------------------------------------------------------
figure()
plot(P_rot(1),P_rot(2), 'mo-', 'markersize', 10,'linewidth', 3); grid minor;
title("Parte b) Rotacion de P en alfa = 21°", "fontSize", 18);
axis([0,5, 0,10])
axis("square")
%-------------------------------------------------------------------------------
```

Resultado:

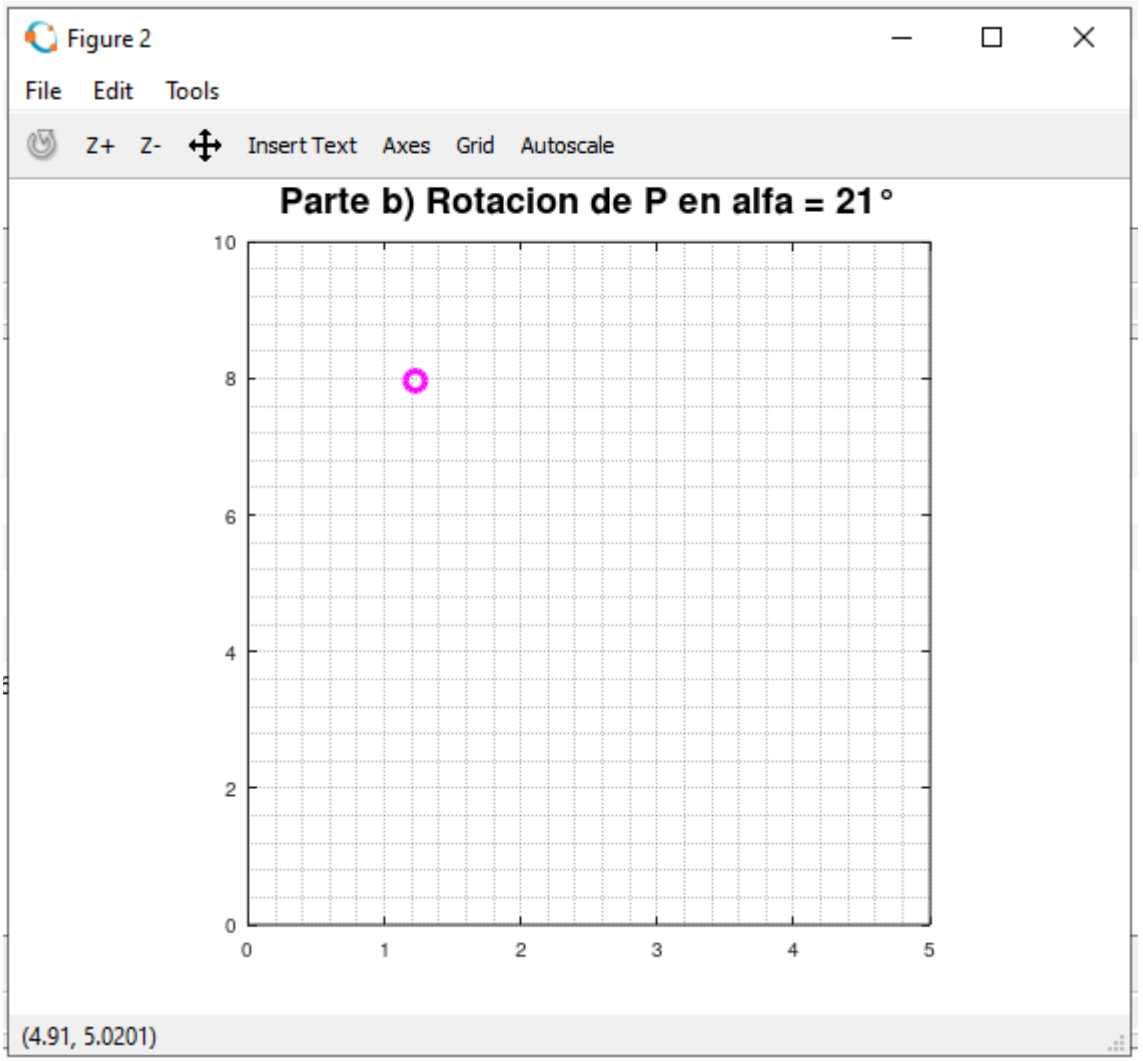


Figure 2
File Edit Tools
Z+ Z- Insert Text Axes Grid Autoscale
Parte b) Rotacion de P en alfa = 21°
10
8
6
4
2
0
0 1 2 3 4 5
(4.91, 5.0201)

c) Al punto resultante del paso anterior, aplíquele una traslación, para desplazarlo una distancia d siguiendo la dirección dada por el vector vd=(1,1).

- Para empezar definiremos el **vector de traslación**, como **d = 26**, **vd = [26.26]** dado que se debía respetar una relación **vd=(1,1)**.
- Luego guardaremos en una variable (**P_r_tra**) la salida de la llamada a **Traslacion2D()** con los parámetros correspondientes.
- Por último graficamos **P_r_tra.**

```
%  Proyecto: Parte 1. Traslación y rotación con elementos matriciales
%  Autor: Andres Romano
%  Fecha: 30 de setiembre de 2021
%  Ultima modificaciÃ³n: 4 de octubre de 2021
clear
clc
%*******************************************************************************
% PARAMETROS:

Origen     = [ 0 ; 0];
Alfa_gra = 21;              % Angulo de rotacion en grados
Vector_d = [ 26 ; 26 ];      % Vector de traslacion
Centro_rot = Origen;          % Punto centro de rotacion

P = [ 4 ; 7 ];
%*******************************************************************************
Alfa_rad = Alfa_gra * (pi / 180); % Convierte grados en radianes
%-------------------------------------------------------------------------------
% Calculo directo de posiciones finales de los movimientos

P_rot = Rotacion2D( P, Centro_rot, Alfa_rad );
P_r_tra = Traslacion2D( P_rot, Vector_d )
%-------------------------------------------------------------------------------
figure()
plot(P(1),P(2), 'mo-', 'markersize', 10,'linewidth', 3); grid minor;
title("Parte a)P = [ 4 ; 7 ];", "fontSize", 18);
axis([0,40, 0,40])
axis("square")
%-------------------------------------------------------------------------------
figure()
plot(P_rot(1),P_rot(2), 'mo-', 'markersize', 10,'linewidth', 3); grid minor;
title("Parte b) Rotacion de P en alfa = 21°", "fontSize", 18);
axis([0,40, 0,40])
axis("square")
%-------------------------------------------------------------------------------
figure()
plot(P_r_tra(1),P_r_tra(2), 'mo-', 'markersize', 10,'linewidth', 3); grid minor;
title("Parte c) Traslacion P rotado en Alfa a distancia d = 26", "fontSize", 18);
axis([0,40, 0,40])
axis("square")
```

Resultado:

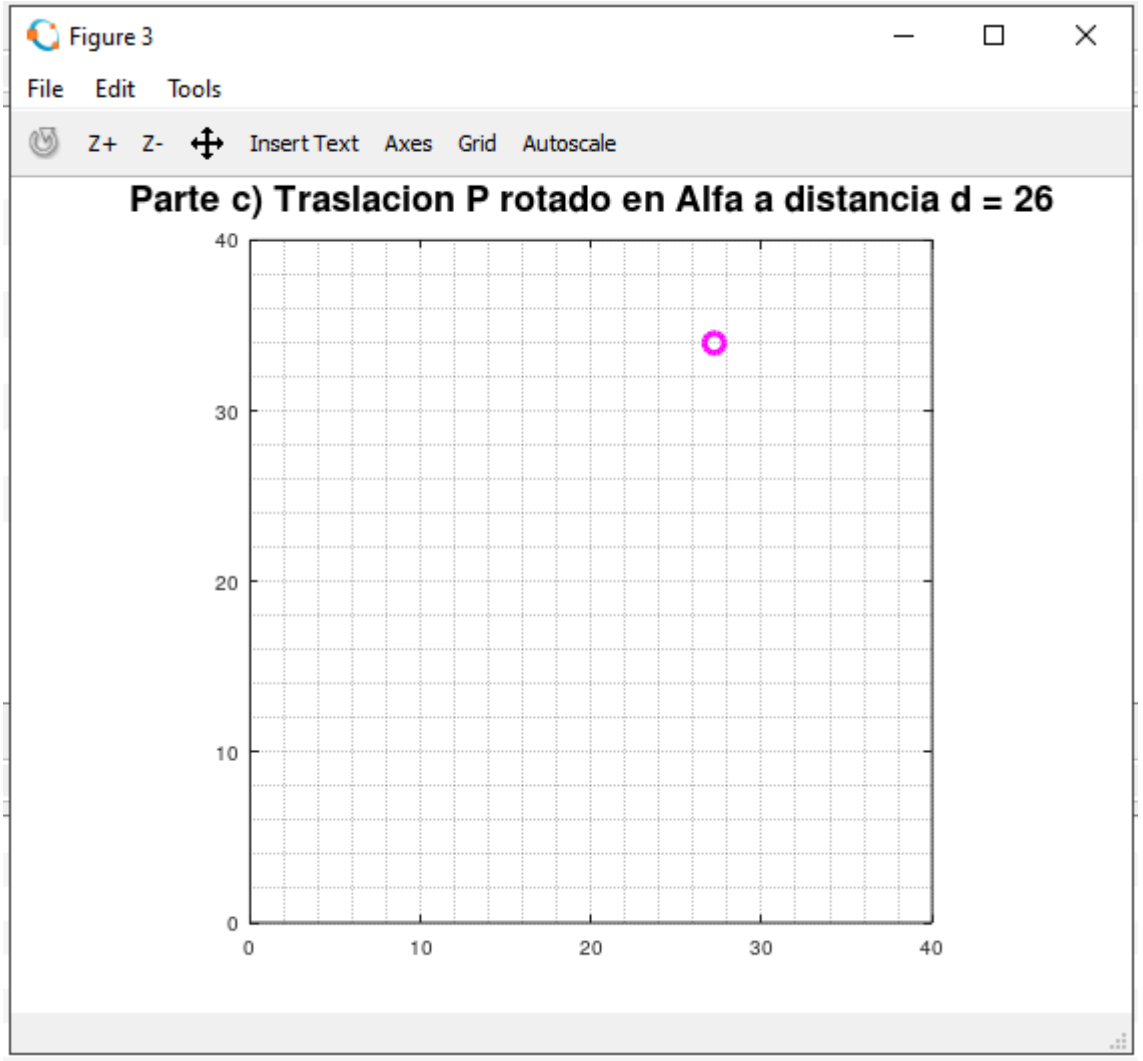


Figure 3
File Edit Tools
Z+ Z- Insert Text Axes Grid Autoscale
Parte c) Traslacion P rotado en Alfa a distancia d = 26
40
30
20
10
0
0 10 20 30 40

d) Presente ahora toda la trayectoria seguida por el punto P, en esos movimientos. Elija la cantidad de puntos intermedios que desee.

```
%  Proyecto: Parte 1. Traslación y rotación con elementos matriciales
%  Autor: Andres Romanc
%  Fecha: 30 de setiembre de 2021
%  Ultima modificaciÃ³n: 4 de octubre de 2021
clear
clc
%*******************************************************************************
% PARAMETROS:

Origen     = [ 0 ; 0];
Alfa_gra = 21;              % Angulo de rotacion en grados
Vector_d = [ 26 ; 26 ];      % Vector de traslacion
Centro_rot = Origen;           % Punto centro de rotacion

P = [ 4 ; 7 ];
Nparte  = 4;                  % Cantidad de partes a dividir los invervalos
%*******************************************************************************
Alfa_rad = Alfa_gra * (pi / 180); % Convierte grados en radianes
%-------------------------------------------------------------------------------
% Calculo directo de posiciones finales de los movimientos

P_rot = Rotacion2D( P, Centro_rot, Alfa_rad );
P_r_tra = Traslacion2D( P_rot, Vector_d );
%-------------------------------------------------------------------------------
% Genera un vector con los puntos relevantes, el de partida y los transformados
vec_tx = [ P(1) P_rot(1) P_r_tra(1) ];
vec_ty = [ P(2) P_rot(2) P_r_tra(2) ];
%-------------------------------------------------------------------------------
[ vr_x , vr_y ] = Rotacion2DTrack(P, Centro_rot, Alfa_rad, Nparte)
%-------------------------------------------------------------------------------
[ vt_x , vt_y ] = Traslacion2DTrack(P_rot, Vector_d, Nparte);
%-------------------------------------------------------------------------------
% Se construye ahora vector que concatena los dos anteriores
%-------------------------------------------------------------------------------
vec_track_x = [P(1) vr_x vt_x];
vec_track_y = [P(2) vr_y vt_y];
%-------------------------------------------------------------------------------
figure()
plot(P(1),P(2), 'mo-', 'markersize', 10,'linewidth', 3); grid minor;
title("Parte a)P = [ 4 ; 7 ];", "fontSize", 18);
axis([0,40, 0,40])
axis("square")
%-------------------------------------------------------------------------------
figure()
plot(P_rot(1),P_rot(2), 'mo-', 'markersize', 10,'linewidth', 3); grid minor;
title("Parte b) Rotacion de P en alfa = 21°", "fontSize", 18);
axis([0,40, 0,40])
axis("square")
%-------------------------------------------------------------------------------
figure()
plot(P_r_tra(1),P_r_tra(2), 'mo-', 'markersize', 10,'linewidth', 3); grid minor;
title("Parte c) Traslacion P rotado en Alfa a distancia d = 26", "fontSize", 18);
axis([0,40, 0,40])
axis("square")

%-------------------------------------------------------------------------------
figure()
plot(vec_track_x, vec_track_y, 'go-', 'markersize', 10,'linewidth', 3); grid minor;
%-------------------------------------------------------------------------------
% Superpone graficos con los puntos principales del trayecto
hold on
plot(vec_tx, vec_ty, 'ro', 'markersize', 16,'linewidth', 3);
title("Parte d) trayectoria conjunta de Rotacion y traslacion", "fontSize", 18);
axis([0,40, 0,40])
axis("square")
%-------------------------------------------------------------------------------
```

Explicación:
Primero definiremos una variable **Nparte** que corresponderá a la cantidad de partes en la que se dividirán los intervalos, en este caso: 4.

```
16  Nparte  = 4;                    % Cantidad de partes a dividir los invervalos
```

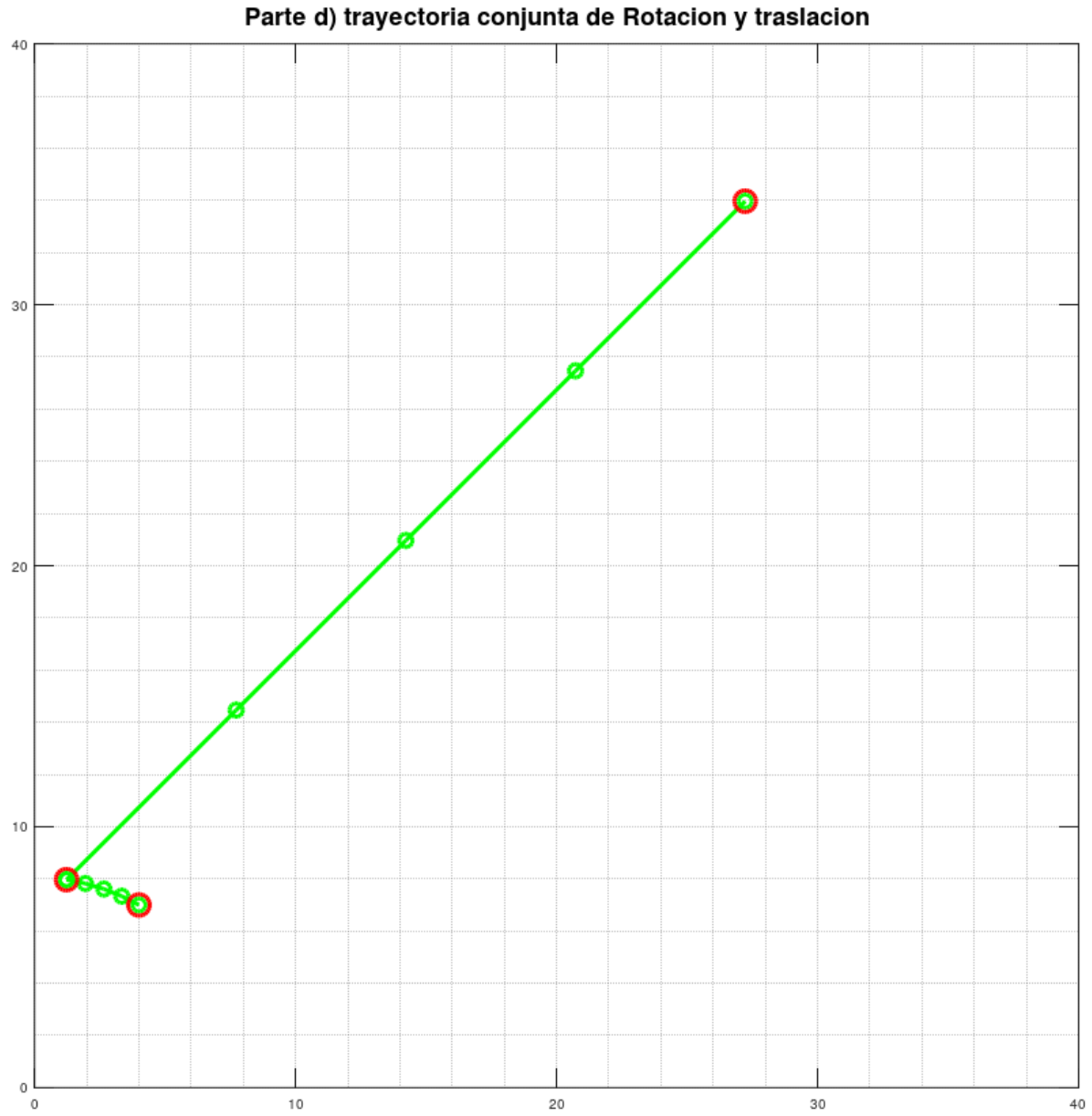
Correspondientemente se agregarán las linea necesarias para generar vectores que contengan los puntos relevantes y otros para guardar los anteriores.

```
24  %-------------------------------------------------------------------------
25  % Genera un vector con los puntos relevantes, el de partida y los transformados
26  vec_tx = [ P(1) P_rot(1) P_r_tra(1) ];
27  vec_ty = [ P(2) P_rot(2) P_r_tra(2) ];
28  %-------------------------------------------------------------------------
29  [ vr_x , vr_y ] = Rotacion2DTrack(P, Centro_rot, Alfa_rad, Nparte)
30  %-------------------------------------------------------------------------
31  [ vt_x , vt_y ] = Traslacion2DTrack(P_rot, Vector_d, Nparte);
32  %-------------------------------------------------------------------------
33  % Se construye ahora vector que concatena los dos anteriores
34  %-------------------------------------------------------------------------
35  vec_track_x = [P(1) vr_x vt_x];
36  vec_track_y = [P(2) vr_y vt_y];
37  %-------------------------------------------------------------------------
```

Para finalizar se grafican los vectores con los puntos anteriores y los transformados.

```
56  %-------------------------------------------------------------------------
57  figure()
58  plot(vec_track_x, vec_track_y, 'go-', 'markersize', 10,'linewidth', 3); grid minor;
59  %-------------------------------------------------------------------------
60  % Superpone graficos con los puntos principales del trayecto
61  hold on
62  plot(vec_tx, vec_ty, 'ro', 'markersize', 16,'linewidth', 3);
63  title("Parte d) trayectoria conjunta de Rotacion y traslacion", "fontSize", 18);
64  axis([0,40, 0,40])
65  axis("square")
66  %-------------------------------------------------------------------------
```

Resultado:


Parte d) trayectoria conjunta de Rotacion y traslacion
40
30
20
10
0
0
10
20
30
40