Optimización de Robots (Mecanismos) utilizando algoritmos evolutivos

Eusebio E. Hernández, Instituto Politécnico Nacional (IPN)

S. Ivvan Valdez, Centro de Investigación en Matemáticas (CIMAT) A.C.

Escuela de Cómputo Evolutivo 2016

1 Introducción

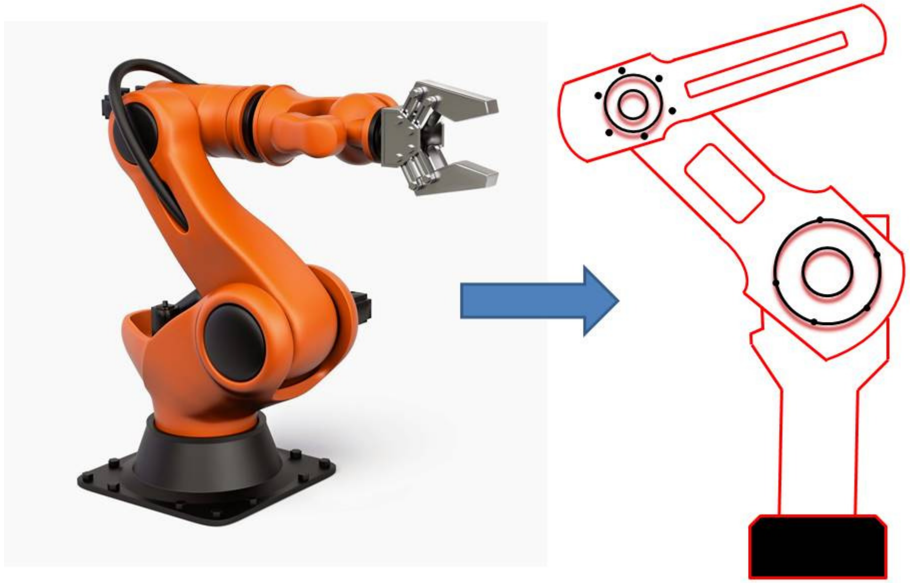


Figure 1: Modelo del brazo robot de 2 grados de libertad

Un **robot manipualdor** es una serie de eslabones inteconectados a través de articulaciones rotacionales o prismáticas en forma de **cadena cinemática abierta** , si el extremo final u órgano terminal donde se coloca la herramienta no se conecta mecanicamente a la base del robot. De otra forma la cadena cinemática formaría un lazo si sus dos extremos están mecanicamente unidos (**robot paralelos**).

2 Cinemática de robots manipuladores

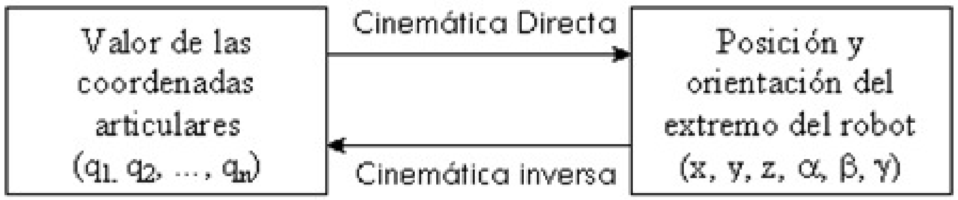
Cinemática es la parte de la física que estudia el movimiento de sistemas mecánicos sin tomar en cuenta las fuerzas que lo originan.

El **modelo de cinemática directa** relaciona las coordenadas articulares $\mathbf{q} \in \mathbb{R}^n$y propiedades mecánicas del robot l_i con las coordenadas cartesianas y orientación de la herramienta en el extremo final $[x, y, z, \theta, \phi, \psi]^T$.

$$(x, y, z, \theta, \phi, \psi) = \mathbf{f}_R(l_i, \mathbf{q}) \tag{1}$$

El **modelo de cinemática inversa** relaciona las coordenadas articulares en función de las coordenadas cartesianas y la orientación de la herramienta en el extremo final del robot manipulador.

$$\mathbf{q} = \mathbf{f}_\mathbf{R}^{-1}(x, y, z, \theta, \phi, \psi) \tag{2}$$



3 Robot planar de 2 grados de libertad

En la Figura 2 se muestra un robot de dos grados de libertad con articulaciones rotacionales que se mueve en el plano vertical $x_0 - y_0$. El sistema de referencia fijo $O_0(x_0, y_0, z_0)$ se considera coincidente con la articulación sobre la base. El ancho de cada servomotor y espesor de la barra metálitca están determinados por β_1 y β_2. Los ejes x_0, y_0 se seleccionan con la regla de la mano derecha.

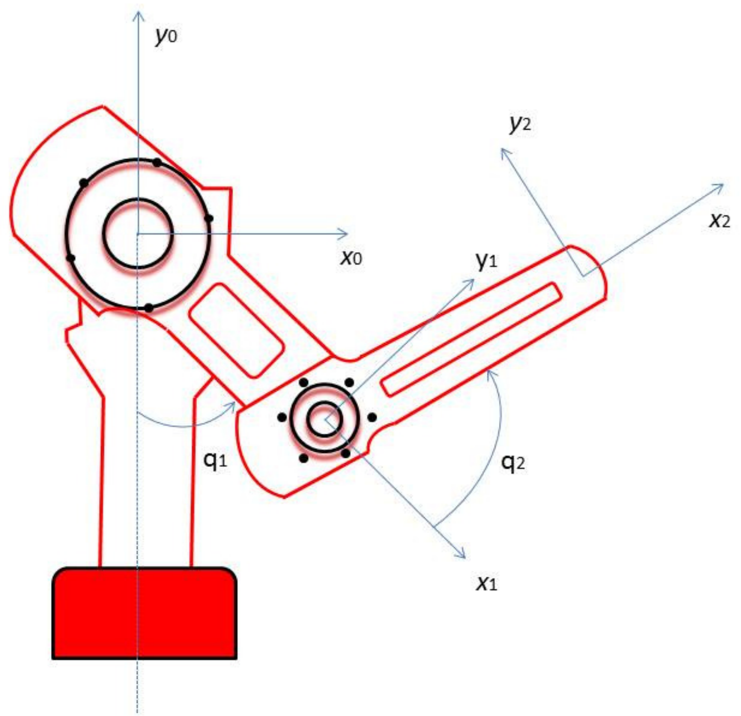


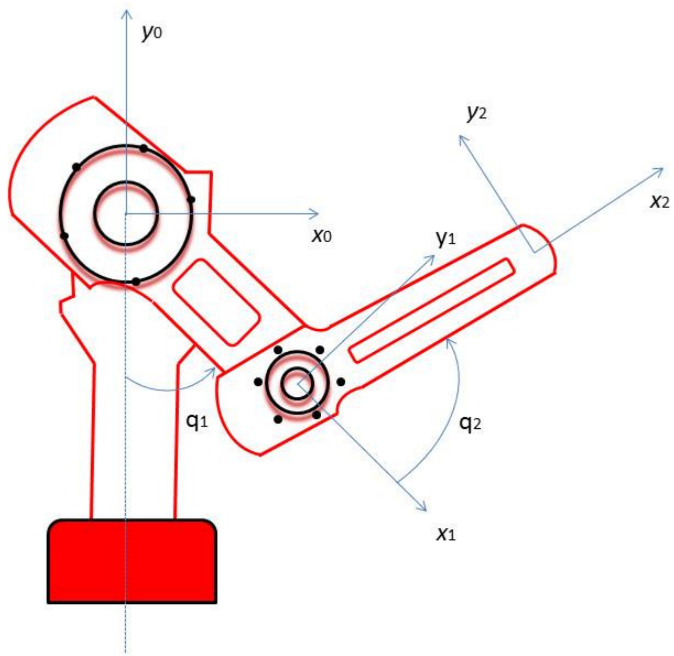
Figure 2: Esquema cinemático del brazo robot

4 Cinemática directa del brazo robot

El sistema de referencia $O_1(x_1, y_1, z_1)$ se coloca en el extremo final del primer eslabón, el eje z_1 se considera paralelo al eje z_0 ($\alpha_1 = 0$). El origen del sistema O_1 se ubica en la intersección del eje x_0 con el eje z_1 (cuando $q_1 = 0$) y está a una distancia β_1 sobre el eje z_0. El origen del sistema de referencia $O_2(x_2, y_2, z_2)$ se considera en el extremo del segundo eslabón, ubicado en la posición del órgano terminal. El ancho del servomotor del codo y el espesor de la segunda barra miden β_2. Cuando $q_2 = 0$ el origen de O_2 se encuentra a una distancia l_2 sobre el eje x_1 y a una distancia β_2 sobre el eje z_1. En la Tabla se muestran los parámetros del brazo robot para la convención Denavit-Hartenberg:

Eslabón	l_i	α_i	d_i	θ_i
1	l_1	0	β_1	q_1
2	l_2	0	β_2	q_2

Table 1: Parámetros Denavit-Hartenberg.



La cinemática directa de las coordenadas cartesianas del órgano terminal del robot (sin considerar su orientación) está dada por:

$$\begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix} = \begin{bmatrix} l_1 sen(q_1) + l_2 sen(q_1 + q_2) \\ -l_1 cos(q_1) - l_2 cos(q_1 + q_2) \\ \beta_1 + \beta_2 \end{bmatrix} \tag{3}$$

Note que para la configuración mostrada, la coordenada z_0 incluye el espesor de los dos servomotores con sus repectivos espesores de las barras metálicas.

5 Cinemática diferencial

La cinemática diferencial es la derivada con respecto al tiempo del modelo de cinemática directa.

$$\frac{d}{dt}\left[x, y, z, \theta, \phi, \psi\right]^T = \begin{bmatrix} \mathbf{v} \\ \mathbf{w} \end{bmatrix}^T = \frac{d}{dt}\mathbf{f}_R(\mathbf{q}) \tag{4}$$

$$\frac{d}{dt}\mathbf{f}_R(\mathbf{q}) = \frac{\partial \mathbf{f}_R(\mathbf{q})}{\mathbf{q}}\dot{\mathbf{q}} = J(\mathbf{q})\dot{\mathbf{q}} \tag{5}$$

El mapeo descrito en términos de la matriz $J(\mathbf{q})$ se denomina Jacobiano del robot y relaciona las velocidades articulares $\dot{\mathbf{q}}$ con la velocidad lineal $\mathbf{v}$ y la velocidad angular $\mathbf{w}$ con las velocidades articulares $\dot{\mathbf{q}}$.

$$J(\mathbf{q}) = \begin{bmatrix} \mathbf{J}_{\mathbf{v}}(\mathbf{q}) \\ \mathbf{J}_{\mathbf{w}}(\mathbf{q}) \end{bmatrix} \tag{6}$$

6 Jacobiano del brazo robot

El Jacobiano del robot de dos eslabones se obtiene como:

$$J(\mathbf{q}) = \frac{\partial \mathbf{f}_R(\beta_1, \beta_2, l_1, l_2, \mathbf{q})}{\partial \mathbf{q}} \tag{7}$$

tomando en cuenta que los parámetros geometricos $\beta_1, \beta_2, l_1, l_2$ son constantes, y considerando la posición *home* del robot sobre el eje y_0

$$J(\mathbf{q}) = \begin{bmatrix} -l_1 sen(q_1) - l_2 sen(q_1 + q_2) & -l_2 sen(q_1 + q_2) \\ l_1 cos(q_1) + l_2 cos(q_1 + q_2) & l_2 cos(q_1 + q_2) \end{bmatrix} \tag{8}$$

El determinante del Jacobiano es $det[J(\mathbf{q})] = l_1 l_2 sen(q_2)$, el cual es cero para $q_2 = 0, +n\pi, -n\pi$. Por tanto, cuando la articulación del codo toma alguno de estos valores para q_2 el brazo robot entra en una *singularidad*.

7 Cinemática inversa

Las ecuaciones de la cinemática inversa del brazo robot se pueden obtener por el siguiente procedimiento considerando el esquema geométrico de la Figura 3.

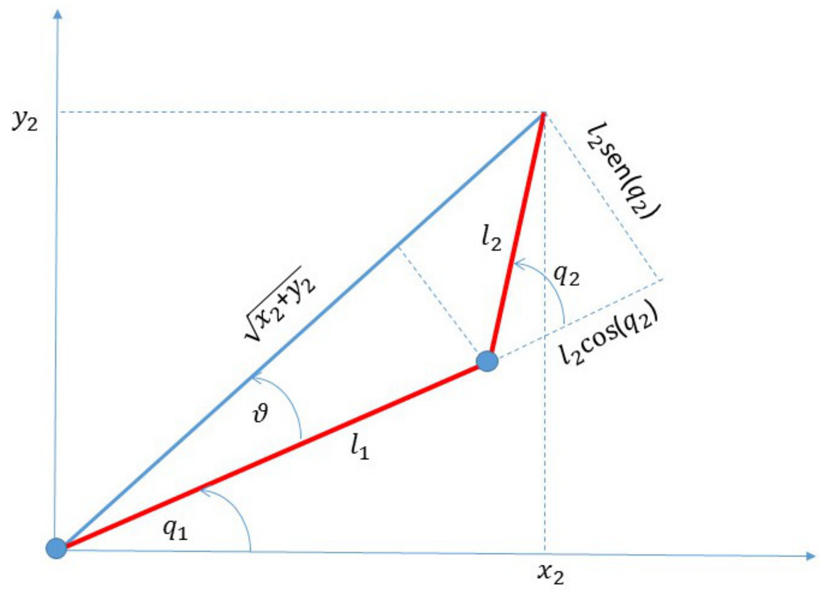


Figure 3: Esquema geométrico del brazo robot

El ángulo ϑ del triángulo formado por los lados adyacente $l_1 + l_2cos(q_2)$, cateto opuesto $l_2sen(q_2)$ y la hipotenusa $\sqrt{x_2^2 + y_2^2}$, del teorema de Pitágoras se obtiene la solución para la variable articular q_2,

$$\begin{aligned} x_2^2 + y_2^2 &= [l_1 + l_2cos(q_2)]^2 + l_2^2sen^2(q_2) \\ &= l_1^2 + l_2^2[cos^2(q_2) + sen^2(q_2)] + 2l_1l_2cos(q_2) \end{aligned} \tag{9}$$

Así,

$$q_2 = cos^{-1}\left(\frac{x_2^2 + y_2^2 - l_1^2 - l_2^2}{2l_1l_2}\right) \tag{10}$$

Note que el angulo ψ satisface

$$\vartheta = tan^{-1}\left(\frac{l_2sen(q_2)}{l_1 + l_2cos(q_2)}\right) \tag{11}$$

Tomando los ángulos $\vartheta + q_1$ dentro del triángulo formado por los catetos adyacente x_f, opuesto y_f y la hipotenusa $\sqrt{x_2^2 + y_2^2}$ se cumple con la siguiente expresión

$$\vartheta + q_1 = tan^{-1}\left(\frac{y_2}{x_2}\right) \tag{12}$$

Entonces la variable articular q_1 toma la siguiente forma

$$q_1 = tan^{-1}\left(\frac{y_2}{x_2}\right) - \vartheta \tag{13}$$

Sustituyendo φ en la ecuación anterior se tiene

$$q_1 = tan^{-1}\left(\frac{y_2}{x_2}\right) - tan^{-1}\left(\frac{l_2 sen(q_2)}{l_1 + l_2 cos(q_2)}\right) \tag{14}$$

En resumen, se tiene el siguiente modelo de cinemática inversa para el brazo robot, el cual no depende de β_1 y β_2:

$$\begin{aligned} q_1 &= tan^{-1}\left(\frac{y_2}{x_2}\right) - tan^{-1}\left(\frac{l_2 sen(q_2)}{l_1 + l_2 cos(q_2)}\right) \\ q_2 &= cos^{-1}\left(\frac{x_2^2 + y_2^2 - l_1^2 - l_2^2}{2 l_1 l_2}\right) \end{aligned}$$

8 Simulación numérica del robot

Existen 3 tipos de problemas, de acuerdo a el tipo de simulación numérica que realicemos con el robot.

- **Simulación de orden 0, o estática**. No involucra velocidades ni información de las masas (inercias) o aceleraciones producidas por la gravedad o los actuadores, para simular el robot solo se utiliza información de la geométrica y el modelo que gobierna el sistema puede ser expresado como una ecuación algebraica (o diferencial de orden 0) .
- **Simulación de orden 1 o cinemática.** Involucra solo las velocidades pero no aceleraciones ni masas, se resuelve una ecuación diferencial de orden 1.
- **Simulación de orden 2 o dinámica.** Se resuelve un modelo dinámico, es decir, que involucra, velocidades , aceleraciones e inercias, la ecuación que gobierna el sistema es diferencial de orden 2.

9 Ejercicio 1. Simulación estática del mecanismo

- **cinemat_dir.m** Dados los ángulos nos devuelve la posición del efector final.
  - Crear un archivo de texto cinemat_dir.m
  - Poner dentro la función como se muestra.
  - En octave mandar llamar la función: l1=0.5; l2=0.5; q1=0.5; q2=0.5; [x2,y2]=cinemat_dir(l1,l2,q1,q2) debe de devolver x2 = 0.66045 y y2 = -0.70894.

$$\begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix} = \begin{bmatrix} l_1 sen(q_1) + l_2 sen(q_1 + q_2) \\ -l_1 cos(q_1) - l_2 cos(q_1 + q_2) \\ \beta_1 + \beta_2 \end{bmatrix}$$

```
% cinemat_dir.m
% Funcion para el calculo de la cinematica directa
function [x2,y2]=cinemat_dir(l1,l2,q1,q2)
  x2=l1*sin(q1)+l2*sin(q1+q2);
  y2=-l1*cos(q1)-l2*cos(q1+q2);
end
```

- **cinemat_inv.m** Dada la posición del efector final nos devuelve los ángulos. Programe la función. **Usar atan2(a,b) para calcular el arco tangente de a/b.** Para [q1,q2]=cinemat_inv(l1,l2,x2,y2); q1=0.5 y q2=0.5

$$\begin{aligned} q_1 &= tan^{-1}\left(\frac{x_2}{-y_2}\right) - tan^{-1}\left(\frac{l_2 sen(q_2)}{l_1 + l_2 cos(q_2)}\right) \\ q_2 &= cos^{-1}\left(\frac{(-y_2)^2 + x_2^2 - l_1^2 - l_2^2}{2 l_1 l_2}\right) \end{aligned}$$

- **det.m** Devuelve el determinante del Jacobiano (para medir manipulabilidad). Programe la función.

$det[J(\mathbf{q})] = l_1 l_2 sen(q_2)$

- ¿Qué pasa si le damos a la cinemática inversa (con l1=0.5 y l2=0.5) el punto x2=0.6, y2=0.7?

- ¿Qué pasa si le damos a la cinemática inversa (con l1=0.5 y l2=0.5) el punto x2=0.6, y2=0.7?
- Si q2 es un imaginario quiere decir que ese punto no es alcanzable por el robot.
- La función isreal(q2) nos devuelve verdadero en caso de que q2 no sea real.
- Modificaremos la cinemática inversa para que q1=1i (un imaginario) en caso de que q2 NO sea real.

10 Maximizar el espacio de trabajo

10.1 Espacio de trabajo

El espacio de trabajo es la región dentro de la cual el efector final del robot puede realizar una tarea.

10.2 Manipulabilidad

La manipulabilidad del robot se refiere a la capacidad del mismo de evitar configuraciones *singulares* o cercanos a la singularidad. Una forma de detectar estas singularidades (y medir la manipulabilidad) es a través del determinante del Jacobiano. En este caso, consideraremos que el robot tiene una buena manipulabilidad si el determinante es mayor a 0.2.

10.3 Funcion objetivo

Se desea maximizar la cobertura del espacio de trabajo con los mínimos largos de los eslabones, sujeto a que el determinante del jacobiano sea mayor que 0.2.

$\max_l \quad ws(l)/(l_1 + l_2 + 1)$

sujeto a

$l_1 + l_2 = 1$

$det(\mathbb{J}(l_1, l_2, q_2)) > 0.2$

11 Función objetivo

Algoritmo para evaluar las dimensiones de un robot dados algunos valores (positivos) para l_1 y l_2 (los parámetros a optimizar).

- Generamos un conjunto de puntos que representan el espacio de trabajo, en este caso de $x = [0.05, 1]$ y $y = [0.05, 1]$, equiespaciados cada 0.05.

```
wy=0.05:0.05:1.0;
n=size(wy,2);
global ws=[]; %el espacio de trabajo se declara global
for wx= 0.05:0.05:1
    ws=[ws; [repmat(wx,1,n); wy]'];
end
scatter(ws(:,1),ws(:,2),'b','filled')
```

- Para cada punto vemos si q_1 y q_2 son valores reales y si el determinante es mayor que 0.2. Y contamos los puntos en que lo anterior se cumple.

- Se calcula $ws = n_r/n_{ws}$ donde n_r es el número de puntos alcanzados (que cumplen las restricciones) y n_{ws} es el número total de puntos en el espacio de trabajo. La evaluación final será $ws/(l_1 + l_2 + 1)$.

15 min para programar la función objetivo:

```
%el espacio de trabajo ws es global, l1=x(1) y l2=x(2)
function [fobj]=funcionObjetivo(x)
```

Archivos cinemat_inv.m, determinante.m y funcionObjetivo.m

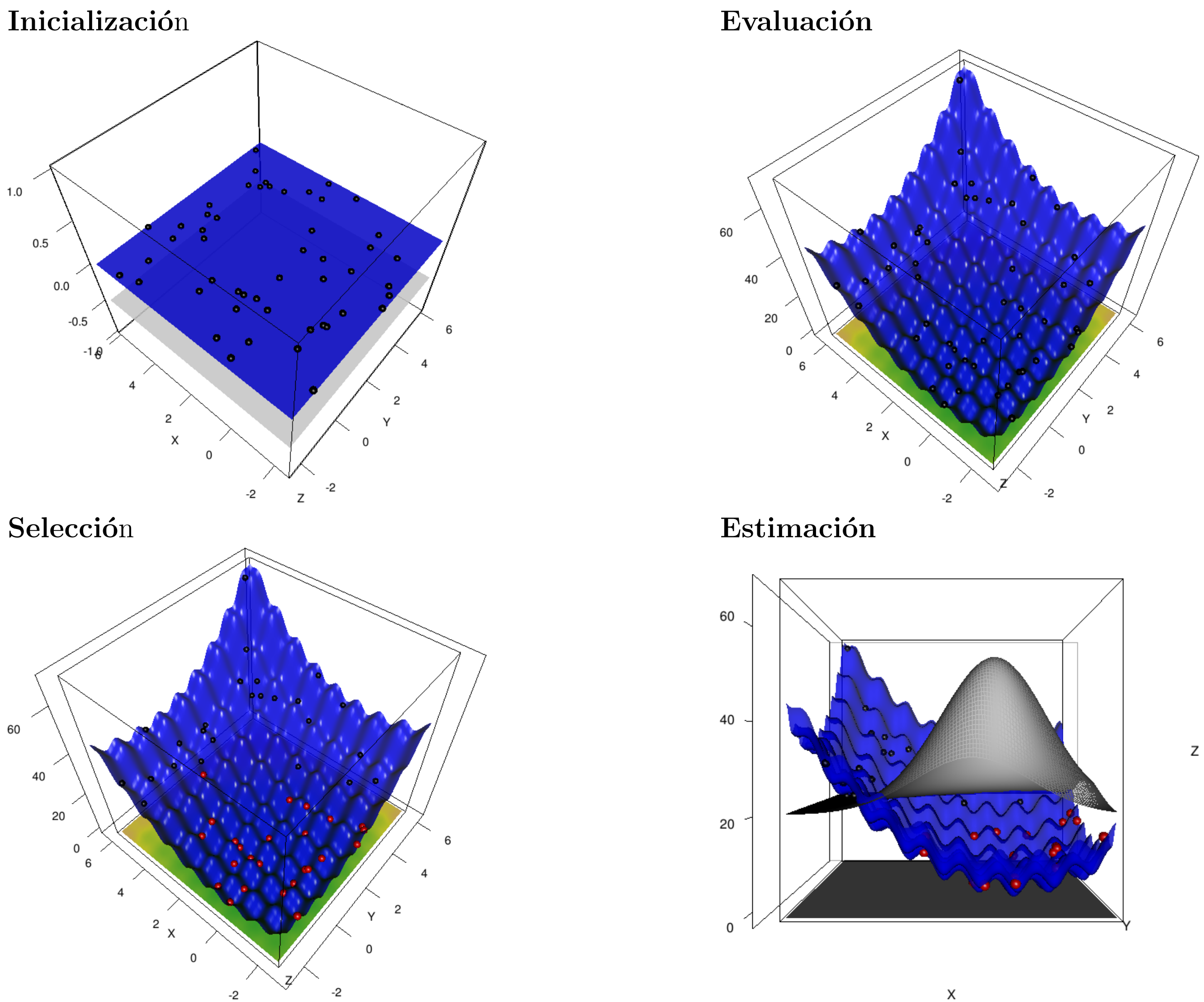
```
function [q1,q2]=cinemat_inv(l1,l2,x2,y2)
  x0=-y2;     y0=x2;
  q2=acos((x0.*x0+y0.*y0-l1*l1-l2*l2)/(2.0*l1*l2));
  if (isreal(q2))
     q1=atan2(y0,x0)-atan2((l2*sin(q2)),(l1+l2*cos(q2)));
  else
     q1=1i;
  end
end
```

```
function d=determinante(l1,l2,q2)
     d=l1*l2*sin(q2);
end
```

```
function [fobj]=funcionObjetivo(x)
global ws; %el espacio de trabajo es global
l1=x(1); l2=x(2);
ncoords=size(ws)(1);
reached=0;
for i=1:ncoords
     wy=ws(i,2);
     wx=ws(i,1);
     [q1,q2]=cinemat_inv(l1,l2,wx,wy);
     if (isreal(q1) && isreal(q2) && determinante(l1,l2,q2)>0.2)
         reached=reached+1;
     end
wsr=reached/ncoords;
fobj=wsr/(l1+l2+1);
end
```

12 Algoritmos de estimación de distribución

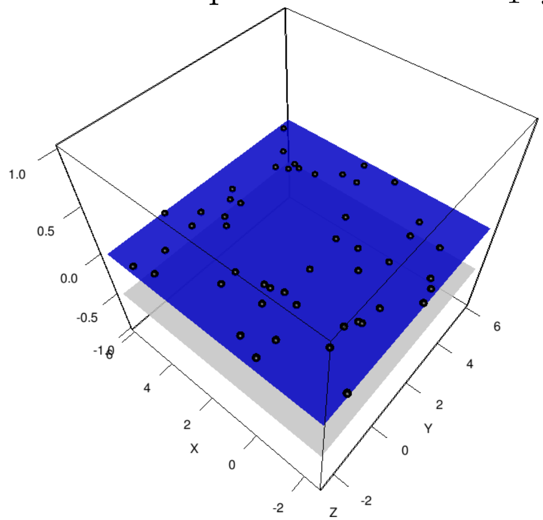
Los EDAs (en inglés) son algoritmos evolutivos que sustituyen los operadores de cruza y mutación por la estimación y muestreo de una distribución de probabilidad.



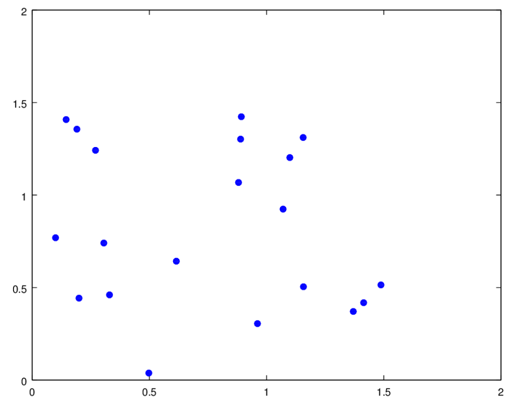
12.1 Inicialización $UMDA_C$

Para inicializar generamos un conjunto de puntos aleatorios Para nuestro caso representarían a l_1 y l_2 .

```
%initialize.m
function [X]=initialize(xinf,xsup,npop,nvar)
    X=[];
    for i=1:nvar
        X=[X, unifrnd(xinf(i), xsup(i), npop, 1)];
    end
end
```



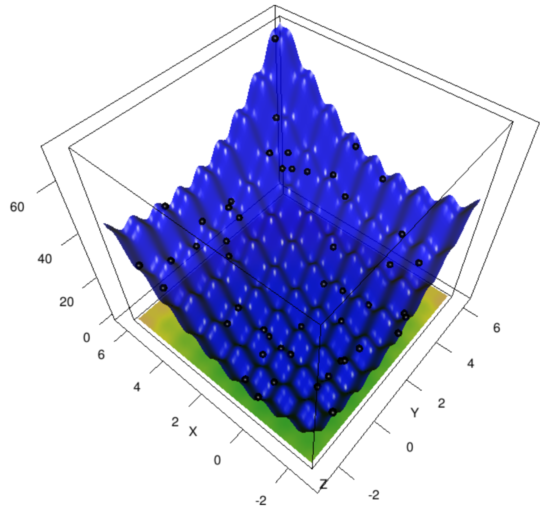

```
%Prueba de la funcion
%definimos los limites de busqueda para l1 y l2
xinf=[0.01 0.01]
xsup=[1.5 1.5]
npop=20
nvar=2 %2 variables a optimizar
X=initialize(xinf, xsup,npop, nvar);
scatter(X(:,1),X(:,2),'b','filled')
```



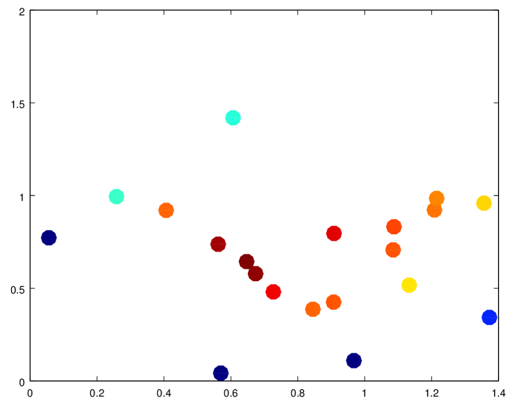
12.2 Evaluación $UMDA_C$

Para evaluar enviamos la función objetivo. Note que en la llamda a la función se considera enviar un vector (sería un vector compuesto por l_1 y l_2)

```
%evaluate.m
function [F]=evaluate(f,X)
npop=size(X,1);
F=zeros(npop,1);
 for i=1:npop
     F(i)=feval(f,X(i,:));
 end
end
```




```
%Prueba de la funcion
%definimos los limites de busqueda para l1 y l2
xinf=[0.01 0.01]
xsup=[1.5 1.5]
npop=20
nvar=2 %2 variables a optimizar
X=initialize(xinf, xsup,npop, nvar);
F=evaluate(”funcionObjetivo”,X);
scatter(X(:,1),X(:,2),15,F,'filled');
```

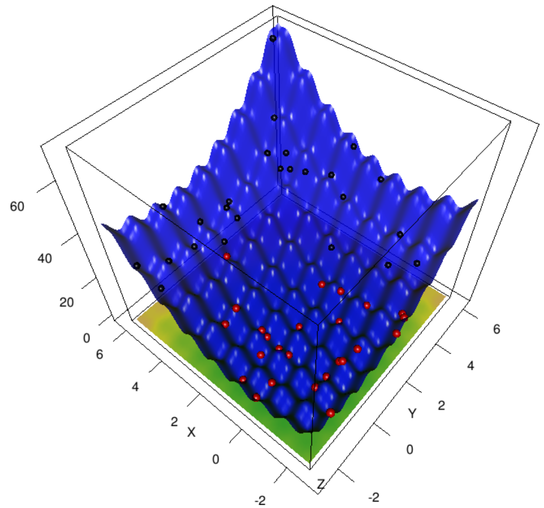


Los rojos son los mejores (mas altos), y los azules los peores(mas bajos).

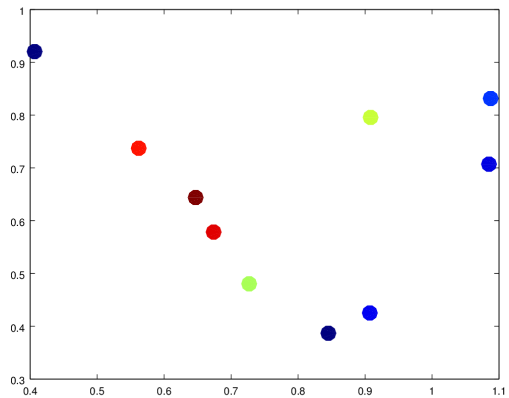
12.3 Selección $UMDA_C$

Para seleccionar tomamos los npop/2 mejores.s

```
%selection.m
function [ix,ibest]=selection(F)
npop=size(F,1);
[Ft,ix]=sort(F,'descend');
ix=ix(1:(npop/2));
ibest=ix(1);
end
```



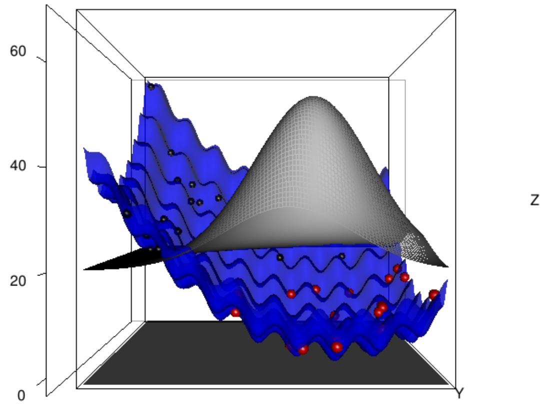

```
%Prueba de la funcion
%definimos los limites de busqueda para l1 y l2
xinf=[0.01 0.01]
xsup=[1.5 1.5]
npop=20
nvar=2 %2 variables a optimizar
X=initialize(xinf, xsup,npop, nvar);
F=evaluate("funcionObjetivo",X);
[ix,ibest]=selection(F);
xbest=X(ibest,:);
fbest=F(ibest);
scatter(X(ix,1),X(ix,2),15,F(ix),'filled');
```



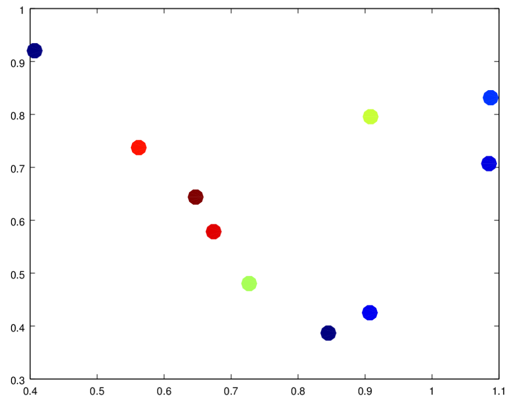
12.4 Estimación $UMDA_C$

Estimamos los parámetros de una distribución normal utilizando el conjunto seleccionado

```
%estimation.m
function [means,sds]=estimation(S)
means=mean(S);
sds=std(S);
end
```




```
%Prueba de la funcion
%definimos los limites de busqueda para l1 y l2
xinf=[0.01 0.01]
xsup=[1.5 1.5]
npop=20
nvar=2 %2 variables a optimizar
X=initialize(xinf, xsup,npop, nvar);
F=evaluate("funcionObjetivo",X);
[ix,ibest]=selection(F);
xbest=X(ibest,:);
fbest=F(ibest);
[means,sds]=estimation(X(ix,:));
scatter(X(ix,1),X(ix,2),15,F(ix),'filled');
```

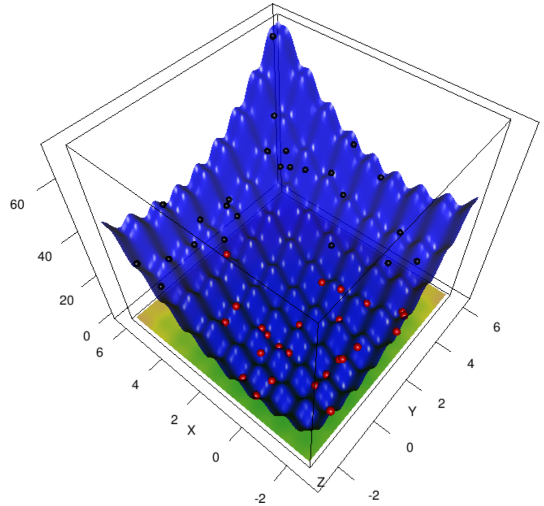


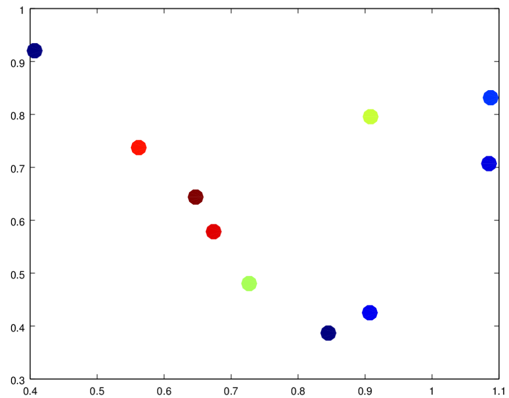
12.5 Muestreo $UMDA_C$

En el muestreo generamos una nueva población, note que generamos npop-1 nuevas soluciones candidatas.

```
%sampling.m
function [X]=sampling(means, sds, npop)
nvar=length(means);
X=[];
 for i=1:nvar
      X=[X, normrnd(means(i), sds(i), npop-1, 1)];
 end
end
```

```
%Prueba de la funcion
%definimos los limites de busqueda para l1 y l2
xinf=[0.01 0.01]
xsup=[1.5 1.5]
npop=20
nvar=2 %2 variables a optimizar
X=initialize(xinf, xsup,npop, nvar);
F=evaluate("funcionObjetivo",X);
[ix,ibest]=selection(F);
xbest=X(ibest,:);
fbest=F(ibest);
[means,sds]=estimation(X(ix,:));
X=[xbest; sampling(means,sds,npop-1)];
scatter(X(:,1),X(:,2),10,'b','filled');
```





12.6 $UMDA_c$

```
%Generacion del espacio de trabajo
%Generacion del espacio de trabajo
wy=0.05:0.05:1.0;
n=size(wy,2);
global ws=[]; %el espacio de trabajo se declara global
for wx= 0.05:0.05:1
  ws=[ws; [repmat(wx,1,n); wy]'];
end
```

```
%UMDA
xinf=[0.01 0.01]; xsup=[1.5 1.5];
npop=20; nvar=2; %poblacion y variables
X=initialize(xinf, xsup,npop, nvar);
F=evaluate("funcionObjetivo",X);
[ix,ibest]=selection(F);
xbest=X(ibest,:);
fbest=F(ibest);
sds=[1000 1000];
scatter(X(:,1),X(:,2),10,F,'filled');
axis([0 2 0 2]);
```

```
%ciclo principal
while(norm(sds)>1e-2)
  [means,sds]=estimation(X(ix,:));
  X=[xbest; sampling(means,sds,npop)];
  F=[fbest; evaluate("funcionObjetivo",X(2:npop,:))];
  [ix,ibest]=selection(F);
  xbest=X(ibest,:);
  fbest=F(ibest);
  disp([xbest fbest]);
  fflush(stdout);
  cla
  scatter(X(:,1),X(:,2),10,F,'filled');
  drawnow()
end
```

13 Paquete de archivos

- Descargar y extraer `www.cimat.mx/~ivvan/public/tallerCE.zip`
- La mejor solución que obtengan la pueden visualizar con plot_sim(l1,l2)

14 Otros ejemplos de diseño optimizado de robots

14.1 Síntesis mecanismo de 4-barras (Caso estático)

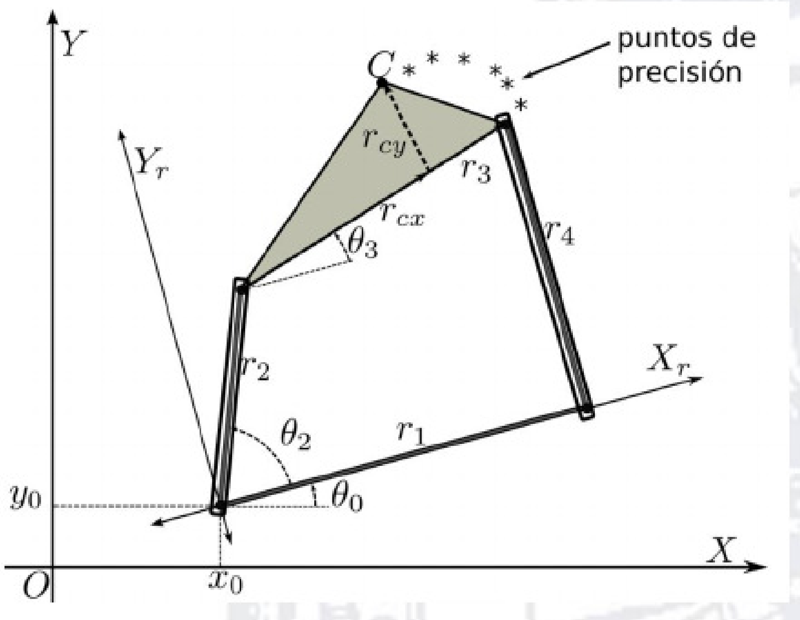


Figura 1. Esquema cinemático de un mecanismo plano de cuatro barras.

El objetivo es que el órgano terminal C pase o se aproxime lo más posible a cada uno de los puntos de precisión.

El par motriz es el relativo a θ_2, dado este valor, la longitud de los eslabones $r_2, r_3, r_4, r_{cx}, r_{cy}$, las coordenadas x_0, y_0 y rotación θ_0 del origen del sistema de coordenadas relativas, es posible determinar las coordenadas del punto C.

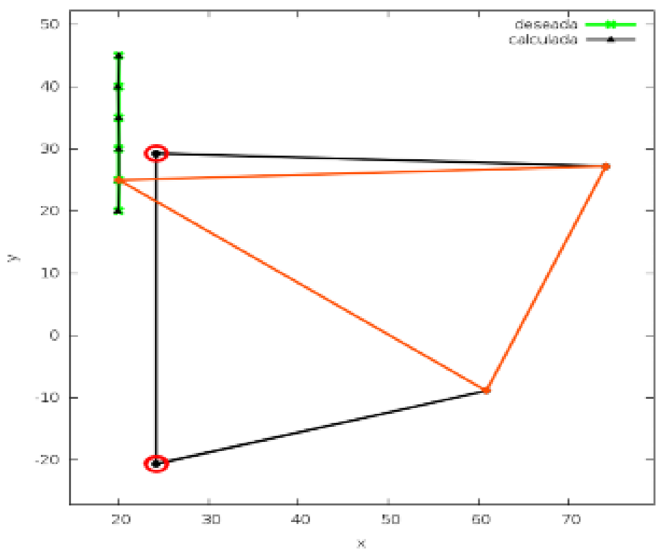


Figura 2. Mejor solución encontrada para el caso 1. Error= 0.000087027, de acuerdo a la Ecuación 11.

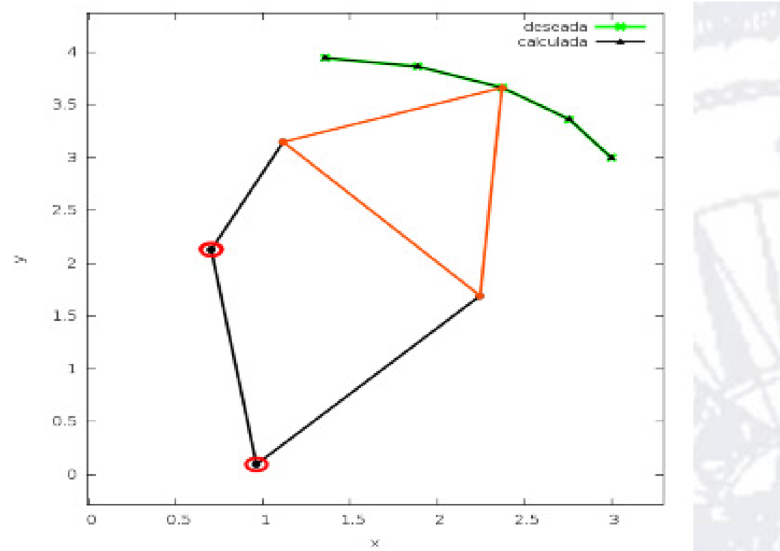


Figura 3. Mejor solución encontrada para el caso 2. Error 0.0000000000010790.

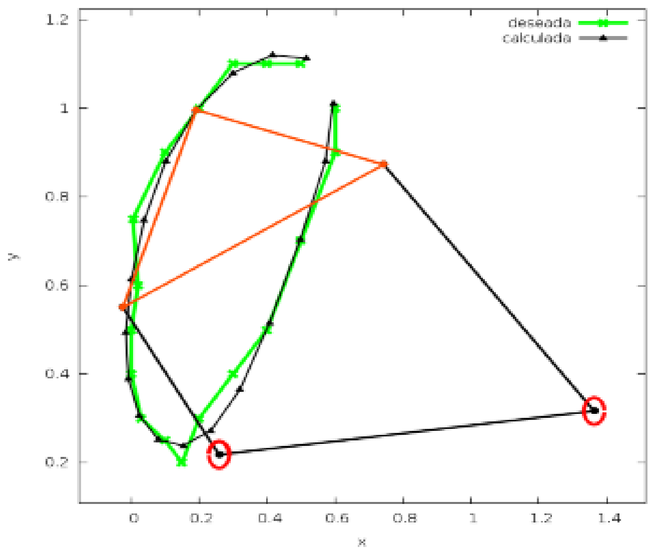


Figura 4. Mejor solución encontrada para el caso 3. Error 0.0000091773883981.

Figure 4: Resultados del proceso de síntesis

Comparativa de resultados

Tabla 3. Comparativo de métodos en la literatura para el problema de síntesis de mecanismos para las trayectorias propuestas por Cabrera et al. (2002)

Algoritmo:	Diferencia Central	Gradiente Exacto	Algoritmo Genético KK	Algoritmo Genético, Cabrera et al. (2002)	EDA Normal
Error Caso 1	No reportado	No reportado	No reportado	0.02617	**0.000087027**
Error Caso 2	6.276e-3	6.8e-5	9.54e-4	1.827e-6	**1.079e-12**
Error Caso 3	2.66e-2	1.68e-2	4.3e-2	2.45e-2	**9.177388e-06**

[1]

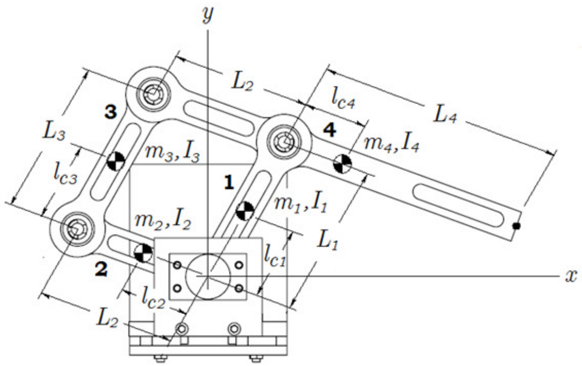
---

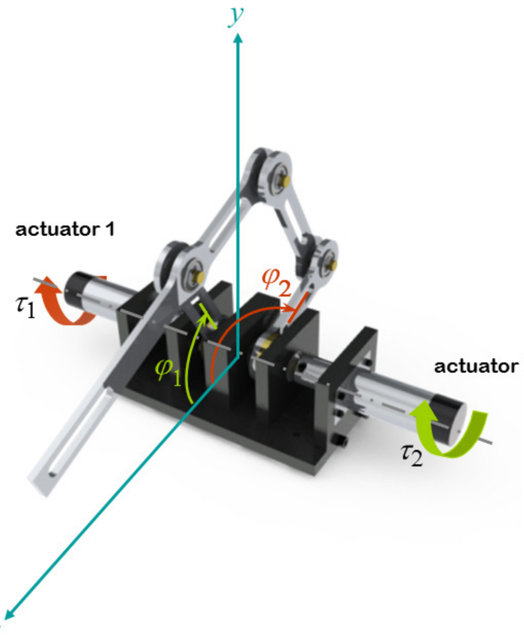
[1]

- Valdez Ivvan , Hernandez Arturo, Botello Salvador, Hernandez Eusebio, "Synthesis of Mechanisms by using an EDA based on the Normal Distribution," Nova Scientia, Vol. 6(1), pp. 45-66, 2013.
- J.A. Cabrera, A. Simon, M. Prado, Optimal synthesis of mechanisms with genetic algorithms, Mechanism and Machine Theory, Volume 37, Issue 10, October 2002, Pages 1165-1177.

14.2 Robot paralelogramo (Caso cinemático)

Ejemplo de optimización estructura-control





Controlador PID para seguimiento de trayectoria minimizando el error $e_i(t)$ en trayectorias

$$u_i(t) = k_p e_i(t) + k_i \int e_i(t)dt + k_d \dot{e}_i(t); \quad i = 1, 2 \tag{15}$$

$$e_i(t) = \varphi_{id} - \varphi_i \tag{16}$$

donde $u_1(t)$ y $u_2(t)$ son los torques τ_1 y τ_2, los cuales controlaran la posición, velocidad y aceleración para los ángulos de entrada φ_1 y φ_2.

$$d_{11}\ddot{\varphi}_1 + \phi_1(\varphi_1) = \tau_1, \tag{17}$$

$$d_{22}\ddot{\varphi}_2 + \phi_2(\varphi_2) = \tau_2. \tag{18}$$

El problema se puede definir como encontrar los tamaños L_1, L_2, L_3 y los parámetros de control: k_1, k_2, k_3, k_4, k_5, k_6. Con la finalidad de diseñar un robot que siga una trayectoria en un tiempo definido.

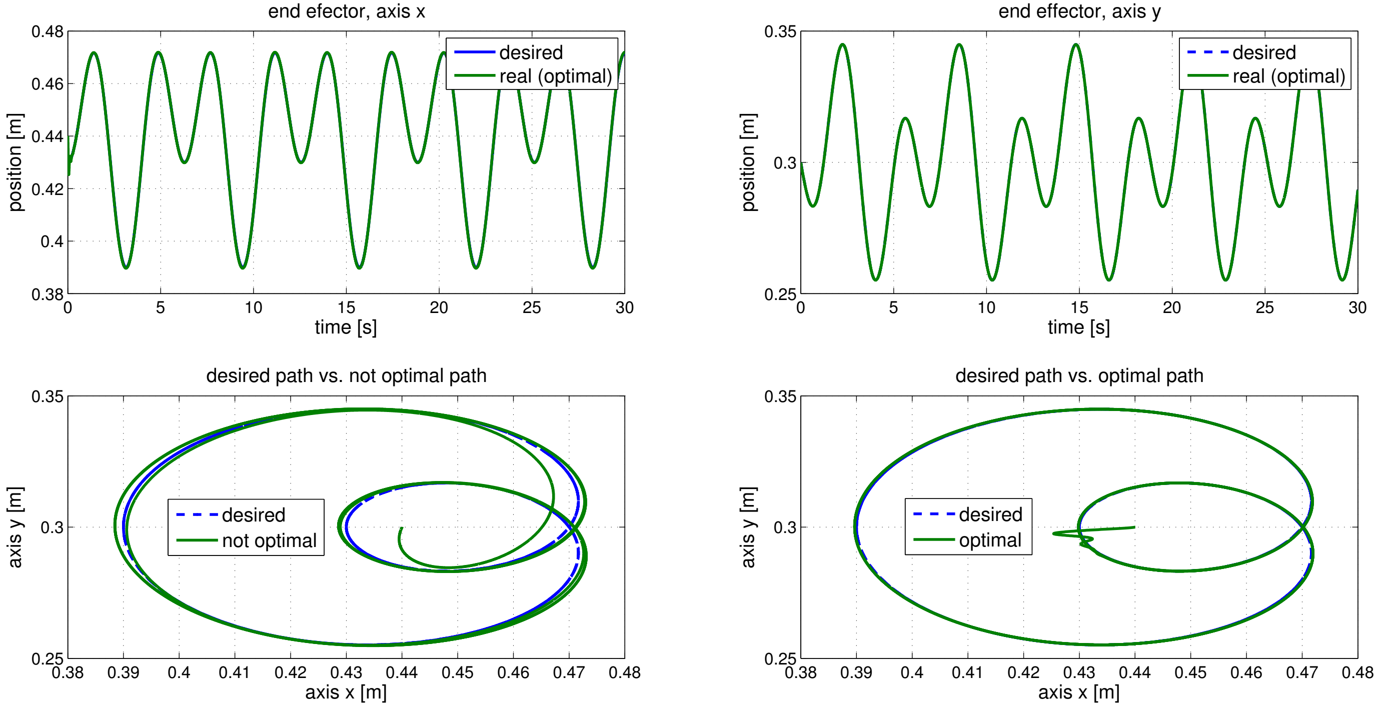


Figure 5: Trayectoria Snail Pascal: deseada *vs.* calculada, sobre el órgano terminal

2

[2] S. Ivvan Valdez and E. Chavez-Conde and Eusebio E. Hernandez and M. Ceccarelli, Structure-control design of a mechatronic system with parallelogram mechanism using an estimation of distribution algorithm, Mechanics Based Design of Structures and Machines, vol. 44(1-2), pp. 58-71,2016, doi: 10.1080/15397734.2015.1035785

14.3 Optimización de un sistema de medición basado en cables

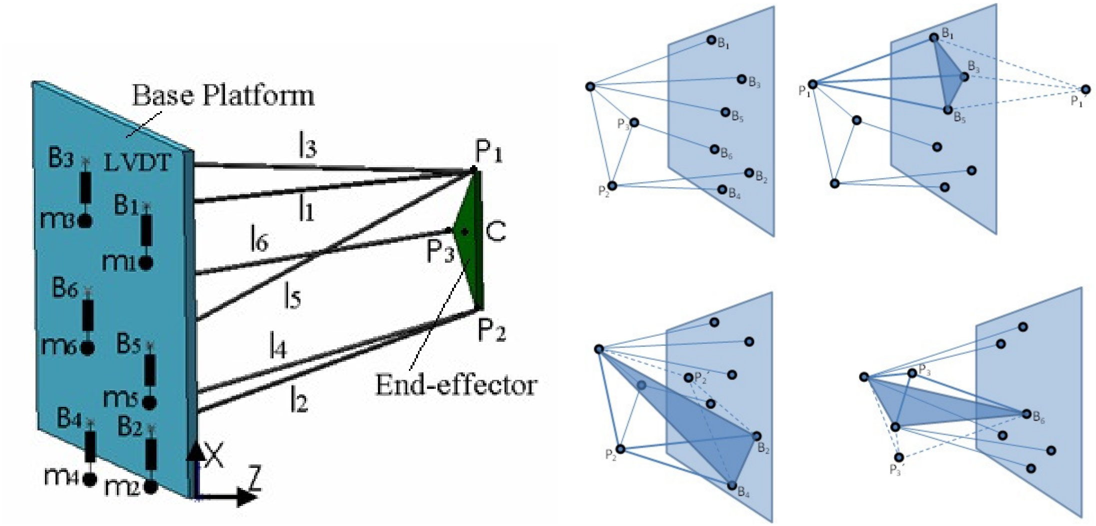


Figure 6: Sistema de medición basado en cables

$$\tilde{p}_4 = \tilde{p}_1 + C_1(-C_2\tilde{v}_1 + C_3\tilde{v}_2 \pm \sqrt{C_4}(\tilde{v}_1 \times \tilde{v}_2)) \tag{19}$$

$$C_1 = \frac{1}{D(\tilde{p}_1, \tilde{p}_2, \tilde{p}_3)} = \frac{1}{\|\tilde{v}_1 - \tilde{v}_2\|}; C_2 = D(\tilde{p}_1, \tilde{p}_2, \tilde{p}_3; \tilde{p}_1, \tilde{p}_3, \tilde{p}_4) = [(\tilde{p}_1 - \tilde{p}_3) \times (\tilde{p}_2 - \tilde{p}_3)] \cdot [(\tilde{p}_1 - \tilde{p}_4) \times (\tilde{p}_3 - \tilde{p}_4)] \tag{20}$$

$$C_3 = D(\tilde{p}_1, \tilde{p}_2, \tilde{p}_3; \tilde{p}_1, \tilde{p}_2, \tilde{p}_4) = [(\tilde{p}_1 - \tilde{p}_3) \times (\tilde{p}_2 - \tilde{p}_3)] \cdot [(\tilde{p}_1 - \tilde{p}_2) \times (\tilde{p}_2 - \tilde{p}_4)]; C_4 = D(\tilde{p}_1, \tilde{p}_2, \tilde{p}_4) \tag{21}$$

La precisión de la estimación de pose (posición y orientación) depende del volumen de los tetrahedros.

El problema de diseño optimizado para el sistema antes descrito se puede definir como: **encontrar la localización de los origenes de los cables sobre la base, con los cuales se maximize el determinante del sistema durante una (o varias) trayectoria completa(s), y evitando tanto como sea posible las intersecciones.**

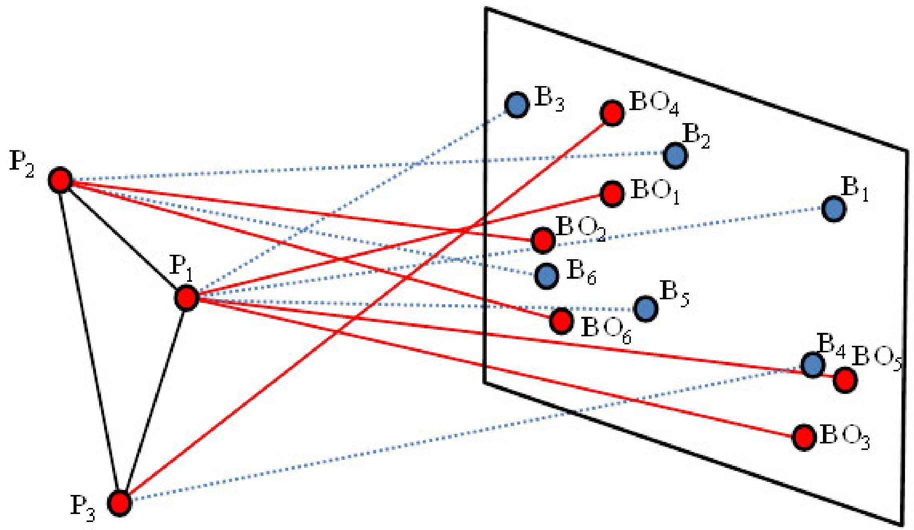


Figure 7: Mejores soluciones para una trayectoria del sistema de medición (rojo), coordenadas originales (azul)

15 Temas de tesis

15.1 Programas de Maestría en el PNPC-CONACYT

- CENTRO DE INVESTIGACIÓN EN MATEMÁTICAS-DEPARTAMENTO DE COMPUTACIÓN
  Maestría en Ciencias de la Computación y Matemáticas Industriales
  Contacto: ivvan@cimat.mx

- INSTITUTO POLITÉCNICO NACIONAL-ESIME TICOMÁN
  Maestría en Ciencias en Ingeniería Aeronáutica y Espacial
  Contacto: euhernandezm@ipn.mx

15.2 Proyectos vigentes relacionados al tema

- FONDO SECTORIAL AGENCIA ESPACIAL MEXICANA-CONACYT
  Modelo de posicionamiento y compensación para la implementación de una plataforma Gough-Stewart general con aplicación a una terminal portátil satelital. Vigente a DICIEMBRE 2018 **(Becas para tesis de Ingeniería y Maestría)**

- FONDO CIENCIA BÁSICA SEP-CONACYT
  Optimización de la estructura-control de robots paralelos basada en algoritmos poblacionales. Vigente al 2 JUNIO 2017.

16 Tarea 1

- Similar al primer trabajo mencionado anteriormente (mecanismo de 4 barras). Se puede dar una trayectoria, y buscar las dimensiones optimas (mínimas) para que el robot pueda pasar por los puntos de precisión.
- Defina una trayectoria circular, dada por un conjunto de puntos (generan puntos sobre una circunferencia).
- Defina una función objetivo que minimice las longitudes del mecanismo que pase por todos los puntos de la que pase por todos los puntos trayectoria anterio. Si un punto NO es alcanzado, debe de ser muy penalizado.
- Ejecute el UMDAc para encontrar las dimensiones óptimas.

17 Tarea 2

- El UMDAc es el algoritmo mas sencillo de los EDAs continuos.
- Una forma de mejorarlo es dar pesos a cada solución en el conjunto seleccionado para el cálculo de los estimadores de media y desviación estandar.
- Cálcule un conjunto de pesos p_i para cada solución i en el conjunto seleccionado que sumen 1 y que cumpla que si $p_i > p_j$ la solución i es mejor que la solución j.
- Calcule estimadores de la media μ y la varianza σ^2 como sigue (note que debe calcular una media y varianza por cada dimensión, es decir una para l_1 y otra para l_2):

  $\mu = \sum p_i \cdot x_i$
  $\sigma^2 = \sum p_i \cdot (\mu - x_i)^2$