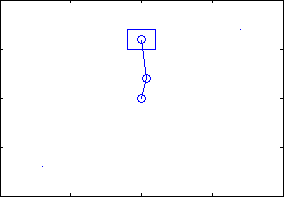
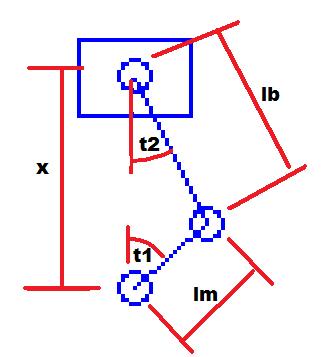
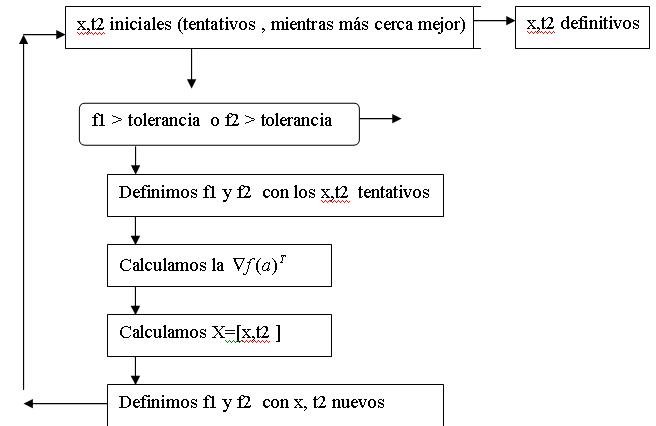
jueves, diciembre 28, 2006

**[Animaciones con Matlab](http://rocoblog.blogspot.com/2006/12/animaciones-con-matlab.html)**

  
Esto puede resultar de ayuda cuando quieres calcular la dinámica de un sistema mecánico, es decir su posición en función del tiempo, su velocidad o su aceleración para el cálculo de las fuerzas de trepidación. Además de el cálculo, uno puede "ver "estas magnitudes, en un animación.  
  
1) Animación en Matlab  
Simplemente es ir calculando para cada instante la posición de nuestro "sistema", las guardamos en matrices o vectores, y después graficamos para cada instante. Luego si se desea se puede guardar en un gif y se ve como el pistón de arriba ( para ver más [Aquí para hacer un gif con matlab](http://rocoblog.blogspot.com/2007/03/como-hacer-un-gift-con-matlab.html))  
  
Aquí el ejemplo... Un piston-biela -manivela:  
  
Tenemos que calcular la dinámica de las piezas, para ello hacemos una maya donde planteamos la geometría.  
  
**Donde**  
**x** es la posición del pistón respecto del eje.  
**lb** es la longitud de la biela  
**lm** longitud de la manivela  
**t1** ángulo entre x y lb  
**t2** ángulo entre lb y x  
  
planteamos la maya:  
  
[](http://www2.udec.cl/~franciscoroco/malla2.jpg)Planteamos el sistema:  
lm.sin(t1) - lb.sin(t2) = 0  
lm.cos(t1) + lb.cos(t2) – x = 0  
  
Son conocidos **lm lb**Como queremos "animar ", nos vamos a dar un valor de t1 (ángulo de la manivela respecto del eje), y calculamos las incognitas t2 y x. El valor de t1 nos lo daremos recorriendo la circunferencia, podemos regular la pausa de la animacion para que funcione con una velocidad angular omega w dada.  
  
  
**¿Cómo resolverlo si es no lineal?**Por medio de la serie de Taylor de la Función e iteración.  
  
Definimos  
F=(f1,f2)  
f1 = lm.sin(t1) - lb.sin(t2) = 0  
f2 = lm.cos(t1) + lb.cos(t2) – x = 0  
  
Resolvemos usando la serie de Taylor de F  
Haciendo T-F=(0,0)  
[mhtml:file://I:\ARCHIVOS%20MATLAB\Archivos%20HTML\RocoBlog%20Animaciones%20con%20Matlab.mht!http://upload.wikimedia.org/math/2/8/d/28d89e02a908cd01774cec68b6223208.png](http://upload.wikimedia.org/math/2/8/d/28d89e02a908cd01774cec68b6223208.png)Como f(a)=0 y a=0  
  
Queda  
[mhtml:file://I:\ARCHIVOS%20MATLAB\Archivos%20HTML\RocoBlog%20Animaciones%20con%20Matlab.mht!http://www2.udec.cl/~franciscoroco/f1.JPG](http://www2.udec.cl/~franciscoroco/f1.JPG)  
Usamos aproximación cortita  
  
Al final queda  
[mhtml:file://I:\ARCHIVOS%20MATLAB\Archivos%20HTML\RocoBlog%20Animaciones%20con%20Matlab.mht!http://www2.udec.cl/~franciscoroco/f2.JPG](http://www2.udec.cl/~franciscoroco/f2.JPG)  
Los valores de X que son X=(x,t2) resultan de resolver esta ecuación, para obtener f1 y f2 nos damos x y t2 tentativos e iteramos… o sea  
  
[](http://www2.udec.cl/~franciscoroco/Dflujo.JPG)  
  
  
  
  
Ahora lo pasamos a matlab. Para hacer todo en orden, hacemos un programa principal, una función para los cálculos en cada instante, para un t1 dado, y una para dibujar el cuadrado del pistón.  
  
  
**1) PROGRAMA PRINCIPAL**  
llama a las funciones y grafica... aqui va la sintaxis  
  
clear all  
%paso del t1  
paso=10\*pi/180;  
%numero de puntos calculados  
nc=(36+1)\*10;  
%valor inicial de t1  
t1=0;  
%estimaciones para t2 y x  
x=70;  
t2=30\*pi/180;  
%definicion de los vectores para guardar los puntos de la maya  
X=[0,0,0];  
Y=X;  
XP=[0,0,0,0,0];  
YP=XP;  
%calculo para cada posicion  
for i=1:nc  
[xx,yy,x,t2] = blogpiston(t1,x,t2);  
X(i,:)=xx;  
Y(i,:)=yy;  
t1=t1+paso;  
%cuadrado para simular piston (Es para que se vea bonito )  
[xx,yy]=cuad(xx(3),yy(3),10,0);  
XP(i,:)=xx;  
YP(i,:)=yy;  
end  
%grafico/animacion  
%Marco para la pantalla  
%misma escala  
%en ambos ejes  
difx=abs(min(X(:,2))-max(X(:,2)));  
dify=abs(min(Y(:,2))-max(Y(:,2)));  
if dify  
d=difx;  
else  
d=dify;  
end  
d=(d\*(1+1/10)+10)\*(1.2);  
for i=1:nc  
hold off  
%Marco para la pantalla  
plot(-d,-d)  
hold on  
plot(d,d)  
%malla  
plot(X(i,:),Y(i,:),'-o')  
%cuadrado piston  
plot(XP(i,:),YP(i,:))  
%tiempo de pausa  
pause(.05)  
end  
  
  
**2) FUNCIÓN**  
La función hace lo del diagrama de flujo es decir dado un t1 calcula la maya para ese t1 en particular. Recordar que se deben guardar las funciones con el nombre correcto, aquí como blogpiston.m  
  
function [xx,yy,x,t2] = blogpiston(t1,x,t2)  
%Tolerancia e  
e=0.001;  
%Dimensiones del sistema  
lm=35;  
lb=56;  
%definicion de F  
f1=lm\*sin(t1)-lb\*sin(t2);  
f2=lm\*cos(t1)+lb\*cos(t2)-x;  
while abs(f1)>e | abs(f2)>e  
df1x=0;  
df1t2=-lb\*cos(t2);  
df2x=-1;  
df2t2=-lb\*sin(t2);  
A=[df1x,df1t2;df2x,df2t2];  
B=[-f1;-f2];  
%resolucion del sistema  
X=A\B;  
%por si acaso no converge  
if X==[0;0]  
break  
end  
%valores nuevos de x y t2  
x=x+X(1);  
t2=t2+X(2);  
%definicion de F con los nuevos x y t2  
f1=lm\*sin(t1)-lb\*sin(t2);  
f2=lm\*cos(t1)+lb\*cos(t2)-x;  
end  
%Puntos de la maya para el t1 dado  
xx=[0,lm\*sin(t1),0];  
yy=[0,lm\*cos(t1),x];  
  
  
  
**3) FUNCION CUADRADO**  
El cuadrado que da belleza el pistón  
Es una función simple que tiene como entrada un punto (x,y) y la mitad del lado de un cuadrado.La función entega los puntos para dibujar un cuadrado al rededor de ese punto. Punto y dibujo.  
  
a ver:  
  
el punto es (x,y)  
e es el lado medio del cuadrado  
a es el angulo en este caso 0  
  
Recordar que se deben guardar las funciones con el nombre correcto, aquí como cuad.m  
  
function [X,Y]=cuad(x,y,e,a);  
r=e\*sqrt(2);  
b=pi/4-a;  
X=[x-r\*sin(b),x+r\*cos(b),x+r\*sin(b),x-r\*cos(b),x-r\*sin(b)];  
Y=[y-r\*cos(b),y-r\*sin(b),y+r\*cos(b),y+r\*sin(b),y-r\*cos(b)];