Trayectoria de un proyectil

Imagina que un cañón que está situado en el origen del plano y que forma un ángulo z con el eje de abscisas, dispara un proyectil con velocidad v(metros/segundo). Las ecuaciones paramétricas de la trayectoria del proyectil, considerando que la aceleración de la gravedad es de 10 m/s^2, vienen dadas por $x(t) = (v\cos z)t$, $y(t) = (v\,\mathrm{sen}\, z)t - 5t^2$.

1 Dadas unas condiciones iniciales v y z, calcula, con la ayuda de comandos simbólicos de *Mathematica*, el tiempo que tarda el proyectil en caer al suelo, el alcance del tiro y la altura máxima; y define las correspondientes funciones de v y z "`tiempo`", "`alcance`" y "`altura`".

Sugerencias. Lo primero que tienes que hacer es informar a *Mathematica* de las funciones que representan la trayectoria del proyectil. Interesa considerar como variables, no sólo el tiempo t, sino también las condiciones iniciales v, z. Así `x[v_,z_,t_]:=v Cos[z] t`. Análogamente defines "`y[v_,z_,t_]`". La función "`tiempo[v_,z_]`" la obtendrás resolviendo respecto a `t` la ecuación $y(v,z,t) = 0$. Ya sé que eso es muy fácil pero se trata de que le pidas a *Mathematica* que lo haga. Para definir "`alcance[v_,z_]:=`" dile a *Mathematica* que calcule `x[v,z,tiempo[v,z]]`. También te recuerdo que `D[y[v,z,t],t]` proporciona la derivada respecto a `t` de `y[v,z,t]`.

Ya está bien de sugerencias. Acaba lo poco que queda de este apartado antes de pasar al siguiente.

2 Haz un gráfico que muestre la trayectoria hasta el instante $t = 5$ de un proyectil disparado con $v = 100$, $z = \frac{\pi}{3}$. Haz que en el gráfico figure un punto rojo indicando la posición del proyectil en el instante $t = 5$.

Sugerencias. Te recuerdo que para dibujar curvas dadas por ecuaciones paramétricas se usa el comando `ParametricPlot[{fx,fy},{t,tmin,tmax},opciones]` donde `fx`, `fy` son funciones de `t` en el intervalo `[tmin,tmax]`. Las `opciones` de `ParametricPlot` son casi las mismas que las de `Plot` las cuales supongo que ya las conoces.

Primero pide a *Mathematica* que dibuje el gráfico de la trayectoria pero no lo muestre en pantalla y asígnale un nombre a este gráfico (`graf1`) para poderlo llamar más tarde.

También necesitarás producir un objeto gráfico: un punto rojo situado en el extremo final de la trayectoria. Usa para ello el comando que produce objetos gráficos cuya sintaxis general, como ya debes saber, es `Graphics[directivas gráficas,primitivas gráficas,opciones]`. Necesitarás sólo dos directivas gráficas: `Hue` (o `RGBColor`) y `PointSize`, y una única primitiva gráfica: `Point[{x,y}]`. No necesitas dar opciones. Asígnale también un nombre a este objeto gráfico (`graf2`).

Como sabes, la salida del comando `ParametricPlot` y de `Graphics` son objetos gráficos que puedes mostrar con el comando `Show[{graf1,graf2,..},opciones]`.

Si antes pusiste `DisplayFunction->Identity` ahora pon `DisplayFunction->$DisplayFunction`.

Bueno, ya lo tienes ¿verdad? Pues ahora viene lo mejor.

3 Aprovecha el trabajo hecho en los puntos anteriores para obtener una función "`disparo[v,z]`" que realice una animación mostrando la trayectoria seguida y el movimiento de un proyectil (representado por un punto rojo) disparado con condiciones iniciales v, z.

Sugerencias. Fíjate, la función "`disparo[v,z]`" debe tener como salida una animación. Como ya sabes, para producir una animación todo lo que se necesita es obtener las gráficas individuales que la forman. En nuestro caso, las gráficas individuales representarán la trayectoria parcial hasta un instante `T`, además deben depender de las condiciones iniciales v, z. En `graf1` sustituye 5 por `T`. También debes sustituir los valores particulares 100 y $\frac{\pi}{3}$ de las condiciones iniciales por v y z. Al resultado puedes llamarle `parabola[v_,z_,T_]`. Como esas parábolas van a formar una animación conviene fijar el intervalo del eje de ordenadas donde se van a visualizar sus gráficas. Usa para ello la opción `PlotRange->{{xmin,xmax},{ymin,ymax}}`. Parece razonable tomar `xmin=0`, `ymin=0`. Como `xmax` es lógico tomar el alcance (que depende de las condiciones iniciales), así `xmax=alcance[v,z]`. Ahora ya sabrás cómo definir `ymax`.
Ahora haces lo propio con `graf2` y le llamas `bala[v_,z_,T_]`. Ya puedes obtener las gráficas individuales modificando convenientemente el comando `Show` del punto anterior. Al resultado puedes llamarle `trayectoria[v_,z_,T_]`.
Finalmente, usa `Table[trayectoria[v,z,T],{T,`T_0`,`T_1`,`$\triangle$`T}]`, donde `[`T_0`,`T_1`]` es el intervalo de variación de `T` que, en principio, parece razonable tomar como `[0,tiempo[v,z]]`, y el incremento de `T`, del que depende el número de gráficas que vas a obtener, puedes tomarlo, para obtener diez gráficas, como $\triangle$`T = tiempo[v,z]/10`.
Bien, ya tienes la función pedida "`disparo[v_,z_]:=`". Quizás precises ajustar algunos pequeños detalles. Comprueba la función con `disparo[100,Pi/3]` y `disparo[100,Pi/4]`.

Para acabar, un regalito. Con `ParametricPlot` puedes conseguir algunas gráficas muy llamativas. Trata de entender lo que sigue.

```
In[1]:=
    lazo[t_]:=ParametricPlot[{Cos[3x],Sin[5x]},{x,0,t},
    PlotRange->{{-1,1},{-1,1}},AspectRatio->Automatic,
    DisplayFunction->Identity,PlotStyle->Hue[t]]

In[2]:=
    punto[t_]:=Graphics[{Hue[0],PointSize[.03], Point[{Cos[3t],Sin[5t]}]}]

In[3]:=
    Do[Show[lazo[t],punto[t],DisplayFunction->$DisplayFunction], {t,1/5,
    2Pi+1/5,1/5}]

In[4]:=
    Table[ParametricPlot[{Cos[3x],Sin[5x]},{x,t,t+1/7},
    PlotRange->{{-1,1},{-1,1}},AspectRatio->Automatic,
    DisplayFunction->Identity,PlotStyle->Hue[t]],{t,0,2Pi,1/7}];

In[5]:=
    Show[%,DisplayFunction->$DisplayFunction]
```

Pues eso es todo, espero que te haya gustado.