# Física con programación

*Luis Fernando Madrid Zapata* *


## 1. Introducción

La Física a pesar de ser una materia práctica requiere para su comprensión de aspectos teóricos que se obtienen mediante el tratamiento matemático de las leyes y principios que rigen la naturaleza. El estudio de estas leyes quedan definidas entonces por fórmulas matemáticas que explican el comportamiento de ciertas variables en algún fenómeno físico determinado. Por fortuna, en los tiempos modernos disponemos de sofisticados equipos que permiten un adecuado análisis experimental de la Física en su parte práctica, además de los computadores que nos abre la posibilidad de efectuar innumerable cantidad de operaciones en décimas de segundo, mediante programas de computador diseñados con lenguajes que le permiten al computador tomar una serie de datos, procesarlos y arrojar resultados.

El presente artículo contiene la codificación en TurboPascal necesaria para el estudio del movimiento parabólico basado en los trabajos presentados por estudiantes del Laboratorio de Física Mecánica. El programa se puede ejecutar desde un sistema operacional con el nombre PARABOLA.EXE requiriendo de archivos BGI y CHR para su ejecución. Quien desee, lo puede adquirir en el Laboratorio de Física Mecánica de la Facultad de Ingeniería de Sistemas de la Universidad Autónoma de Manizales.

## 2. El problema físico y el algoritmo

Cuando nos enfrentamos a un problema físico nos cuestionamos las siguientes inquietudes:

1. Cuáles son los datos o condiciones que conocemos del problema?
2. Cuáles son las leyes o principios que entran en juego?
3. Cuáles son los resultados obtenidos?

Las tres preguntas planteadas corresponden con las de un algoritmo de programación, por lo que el problema se puede tratar como tal.

* *Profesor Universidad Nacional, Universidad Autónoma*

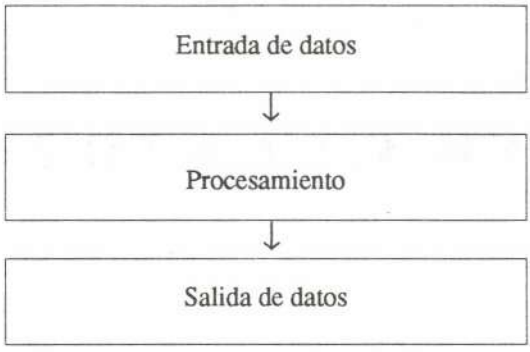


Figura 1. Algoritmo de programación

*...para resolver un problema físico, es necesario conocer por lo menos un lenguaje de programación que nos permita el acceso al computador y al sinfín de operaciones que puede realizar.*

Para que el computador pueda recibir datos de entrada, procesarlos y entregar los resultados correspondientes, se requiere de un lenguaje de programación. En la actualidad encontramos una gran cantidad de lenguajes y paquetes de programación en el mercado que son de fácil consecución y fácil manejo; lenguajes como el Basic, el TurboPascal, el Turbo C, el Fortran, entre otros, nos dan la posibilidad de procesar unos datos de entrada y obtener resultados por medio de una codificación adecuada.

De esta codificación se encarga el Ingeniero de Sistemas a quien le corresponde la labor de organizar una serie de pasos de programación lógicos y acordes con el problema que se plantea para obtener resultados de un modo preciso y eficiente.

**3. El lenguaje**

Consiste en códigos, señales, comandos que al ser recibidos por el computador, este efectúa una función o una instrucción que la comprende perfectamente si el programa está bien codificado. Para asegurarse de que una codificación es correcta el programador dispone de un compilador que le especifica cuales son las fallas que puede presentar el programa y efectuarles su respectiva corrección. Por lo tanto para resolver un problema físico, es necesario conocer por lo menos un lenguaje de programación que nos permita el acceso al computador y al sinfín de operaciones que puede realizar.

**4. Movimiento parabólico**

Es un movimiento que resulta de la combinación de otros dos movimientos, uno horizontal que es uniforme, y uno vertical que es una caída libre, es decir, con aceleración constante e igual a la aceleración de la gravedad. Conociendo la velocidad inicial, el ángulo inicial, la posición inicial y la altura inicial podemos obtener por los principios del movimiento de los

cuerpos diferentes datos como la máxima altura alcanzada por el objeto, la máxima distancia horizontal antes de llegar de nuevo a tierra, la velocidad o la posición en cualquier instante de tiempo, la trayectoria descrita por el objeto.

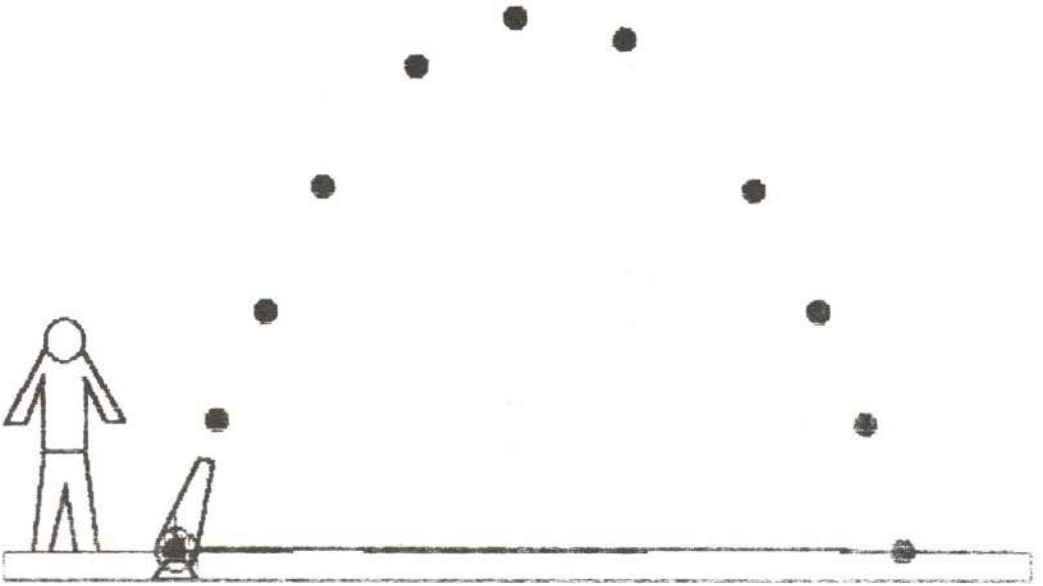

Figura 2. Movimiento parabólico.

El programa que se detalla a continuación lo pueden obtener en la sala de micros de la Universidad Autónoma de Manizales. Este permite todos estos cálculos además de una ilustración gráfica que aproxima el análisis a una situación real en la que es disparada una bala de cañón con condiciones que el usuario puede modificar a su gusto por medio de un menú de opciones. A continuación de cada paso de programa se explican las instrucciones del TurboPascal empleadas; es necesario tener presente que las instrucciones y procedimientos no emplean letras tildadas :

```
Uses crt,graph;
Var c,ango,m,n,r,esc,i,color,tar,mopa,opcion:integer;
   x,y,yo,ang,g,vo,v,vx,vy,t,incre,xm,ym,tv,vari:real;
```

Uses: declaración del uso de unidades compiladas separadamente.
crt: unidad estandard que permite la escritura de textos por pantalla.
graph: unidad estandard que permite la construcción de gráficas por pantalla.
Var: declaración de variables que se van a emplear en el programa.
integer: tipo escalar de datos que corresponde a los números enteros comprendidos entre -32768 y 32767.

real: tipo escalar de datos que corresponde a los números reales que están comprendidos entre $2.9*10^{-39}$ y $1.7*10^{38}$ positivos y negativos.

```
Procedure iniciar(z1,z2,z3:string);
   begin
      initgraph(tar,mopa,'');
      rectangle(40,40,getmaxx-40,getmaxy-40);
       setfillstyle(1,1);
      Floodfill(getmaxx div 2, getmaxy div 2,15);
      bar3d(60,60,getmaxx-60,getmaxy-60,10,true);
      setfillstyle(9,2);
      floodfill(getmaxx div 2,getmaxy div 2,15);
      settextstyle(0,0,5);
      setcolor(4);
      outtextxy(120,100,'MOVIMIENTO');
       outtextxy(120,150,'PARABOLICO');
      settextstyle((0,0,2);
      setcolor(4);
      outtextxy(150,250,z1);
      settextstyle(0,0,3);
      setcolor(4);
      outtextxy(60,300,z2);
      settextstyle(0,0,2);
      setcolor(4);
      outtextxy(80,350,z3);
      outtextxy(350,400,'presione ENTER');
      readln;clearviewport
   end;
```

Procedure iniciar(z1,z2,z3:string): procedimiento establecido por el usuario que consta de un encabezamiento que contiene la palabra reservada Procedure seguida por un identificador que se constituye en el nombre del procedimiento (iniciar), opcionalmente seguida por una lista de parámetros formales ((z1,z2,z3:string)). A continuación viene un bloque que a su vez se compone de una parte de definiciones, declaraciones y una parte de proposiciones.

begin - end: palabras reservadas que encierran un bloque de proposiciones.
initgraph(tar,mopa,''): procedimiento que carga el Hardware gráfico, carga a la memoria el manejador apropiado de la tarjeta graficadora y coloca el sistema en el modo gráfico deseado. El parámetro tar corresponde a una variable entera, cuyo valor determina la tarjeta graficadora. El parámetro mopa es una variable entera que envía el modo gráfico así como la paleta de colores. El parámetro '' es una cadena de caracteres que en este caso corresponde a la cadena vacía e indica que el manejador de la tarjeta se encuentra en el directorio por omisión.

rectangle(40,40,getmaxx-40,getmaxy-40): es usada para dibujar un rectángulo con el color y estilo de la línea corriente. La esquina superior izquierda del rectángulo está determinada por las dos primeras coordenadas y la esquina inferior derecha corresponde a las dos últimas coordenadas.
getmaxx: regresa el valor de la columna más a la derecha del modo gráfico corriente.
getmaxy: regresa el valor de la fila más al fondo en el modo gráfico corriente.
setfillstyle(1,1): selecciona el estilo de llenado de figuras cerradas mediante dos variables enteras, la primera selecciona el tipo de llenado y la segunda variable escoge el color para el llenado.
floodfill(getmaxx div 2, getmaxy div 2,15): permite llenar con un color una figura cerrada. Las dos primeras variables corresponden a las coordenadas de un punto dentro de la figura. Si el punto se coloca fuera de la figura, se pinta la parte externa de ella. La tercera variable especifica el color del borde de la figura que se desea pintar. Si el borde no coincide con el color de la figura, el llenado se sale de ella.
div: es un operador similar a la división solo que entrega la parte entera del cociente.
settextstyle(0,0,5): por medio de esta instrucción se envía el tipo de letra, la dirección y el tamaño del texto gráfico, mediante la determinación de tres parámetros, el primero corresponde a los tipos de letras soportados (texto simple, texto triple, texto gótico, etc.), el segundo determina si el texto se escribe horizontal (0) o verticalmente (1), el tercero indica el tamaño soportado por el texto y comprende valores 1..10.
setcolor(4): selecciona el color de las líneas y del texto gráfico.
outtextxy(120,100,'MOVIMIENTO'): coloca la cadena correspondiente al tercer parámetro a partir del punto de coordenadas dadas por las dos primeras variables.
readln: espera la lectura de una variable.
clearviewport: permite borrar la pantalla gráfica o la ventana gráfica activa previamente seleccionada.

```
Procedure Calcular(t:real);
    begin
        Vx:=Vo*Cos(Ango*Pi/180);
        Vy:=Vo*Sin(Ango*Pi/180)-g*t;
        V :=sqrt(sqr(Vx)+sqr(Vy));
        X :=Vx*t+Xo;
        Y :=-1/2*g*sqr(t)+Vo*Sin(Ango*Pi/180)*t+Yo;
        if Vx<>0 then Ang:=arctan(Vy/Vx)*180/Pi else Ang:=90;
        setcolor(5);
        if Vy>0 then
        setviewport(6*r+trunc(x*5*r)+10, getmaxy-trunc(y*5*r)-r+20,
            6*r+trunc(x*5*r)+91,getmaxy-trunc(y*5*r)-r+91, true);
    else
        setviewport(trunc(x*5*r)+10, getmaxy-trunc(y*5*r)-r+10,
            trunc(x*5*r)+91,getmaxy-trunc(y*5*r)-r+91, true);
     clearviewport;
```

```
    rectangle(0,0,65,70);
    str(t:5:2,k);outtextxy(1,1,'t='+k);
    str(x:5:2,k);outtextxy(1,10,'x='+k);
    str(y:5:2,k);outtextxy(1,20,'y='+k);
    str(Vx:5:2,k);outtextxy(1,30,'Vx='+k);
    str(Vy:5:2,k);outtextxy(1,1,'Vy='+k);
    str(V:5:2,k);outtextxy(1,1,'V='+k);
    str(Ang:5:2,k);outtextxy(1,1,'An='+k);
     setviewport(0,0,getmaxx,getmaxy,true)
End;
```

Vx:=Vo*Cos(Ango*Pi/180): asigna a la variable que aparece a la izquierda del signo := el resultado numérico de las operaciones que se indican a la derecha.
if Vx<>0 then Ang:=arctan(Vy/Vx)*180/Pi else Ang:=90: proposición condicional que determina que si la condición Vx<>0 es verdadera entonces deberá ejecutarse la proposición Ang:=arctan(Vy/Vx)*180/Pi y en caso contrario deberá ejecutarse la proposición Ang:=90.
Las condiciones que se pueden presentar son:

| | | | |
|---|---|---|---|
| = | igual | <> | diferente |
| < | menor que | > | mayor que |
| <= | menor o igual | >= | mayor o igual. |

setviewport(0,0,getmaxx,getmaxy,true): activa una ventana gráfica en la que los dos primeros parámetros son las coordenadas de la esquina superior izquierda y los dos siguientes parámetros son las coordenadas de la esquina inferior derecha, la última variable es de tipo booleano. Si esta toma el valor TRUE, la parte de la gráfica que no quepa en la ventana, no se mostrará en pantalla. Si toma el valor de FALSE , cuando la gráfica sea más grande que la ventana, esta gráfica se mostrará usando parte de la pantalla que no pertenece a la ventana.
str(t:5:2,k); procedimiento que convierte el valor numérico de la variable t tomando 5 dígitos de los cuales 2 son decimales en un valor de tipo string y almacena el resultado en la variable k.
trunc(x*5*r): toma la parte entera del valor que se obtiene con las operaciones dentro del paréntesis.

```
Procedure bola(a,b:integer);
   begin
      setcolor(3);
       circle(a,getmaxy-b,r div 2);
       setfillstyle(1,1);
      floodfill(a,getmaxy-b,3);
       delay(trunc(incre*500));
      setfillstyle(0,0);
       floodfill(a,getmaxy-b,3);
      setcolor(c);
       circle(a,getmaxy-b,r div 2)
   End;
```

circle(a,getmaxy-b,r div 2): dibuja un círculo con centro en el punto cuyas coordenadas las definen los dos primeros parámetros, cuyo radio lo determina la tercera variable y el color está determinado por la instrucción setcolor.

delay(trunc(incre*500)): da un tiempo de espera para efectuar la siguiente instrucción de acuerdo con el valor entre paréntesis que está dado en milisegundos.

```
Procedure canon(m,n:integer; angulo:real);
    var a,b,c,d,e,f:integer;
    begin
        angulo:=angulo*pi/180;
        line(m,getmaxy-n,m+r,getmaxy-n+r);
        line(m+r,getmaxy-n+r,m-r,getmaxy-n+r);
        line(m-r,getmaxy-n+y,m,getmaxy-n);
        circle(m,getmaxy-n,r);
        a:=trunc(m-r*sin(angulo));
        b:=trunc(getmaxy-n-r*cos(angulo));
        c:=trunc(m+5*r*cos(angulo)+r div 3*sin(angulo));
        d:=trunc(getmaxy-n-5*r*sin(angulo)+r div 3*cos(angulo);
        line(a,b,c,d);
        a:=trunc(m+r*sin(angulo));
        b:=trunc(getmaxy-n+r*cos(angulo));
         e:=trunc(a+5*r*cos(angulo)-r div 3*sin(angulo));
         f:=trunc(b-5*r*sin(angulo)-r div 3*cos(angulo));
        line(a,b,e,f);
        line(c,d,e,f);
    end;
```

line(a,b,c,d):esta instrucción dibuja una línea desde el punto de coordenadas (a,b) hasta el punto de coordenadas (c,d) usando por omisión el color 3 de la paleta activa o el color enviado por la instrucción setcolor. El estilo de la línea y su repintado está definido por la instrucción setlinestyle.

```
Procedure giro_canon;
    var i,o:integer;
     procedure giro;
    begin
        sound(50+i);
        setcolor(3);setfillstyle(1,5);
        canon(m+r,n+r,2*i);
        floodfill(m+r,getmaxy-r div 2-n,3);
        setfillstyle(1,14);
        floodfill(trunc(m+r+3*r*cos(2*i*pi/180)),trunc(getmaxy-r-
                        n-3*r*sin(2*i*pi/180)),3);
```

```
        setfillstyle(0,0);
        floodfill(trunc(m+r+3*r*cos(2*i*pi/180)),trunc(getmaxy-r-
                        n-3*r*sin(2*i*pi/180)),3);
        setcolor(0);
        canon(m+r,n+r,2*i);
        floodfill(trunc(m+r+3*r*cos(2*i*pi/180)),trunc(getmaxy-r-
                        n-3*r*sin(2*i*pi/180)),0)
    end;
begin
    if ango>0 then o:=ango+4 else
    if ango=0 then o:=0 else ango:=ango-4;
     if ango>0 then for i:=0 to o div 2 do giro
              else for i:=0 downto o div 2 do giro;
    setfillstyle(1,5);
    setcolor(3);
    canon(m+r,n+r,o);
    floodfill(trunc(m+r+3*r*cos(2*i*pi/180)),trunc(getmaxy-r-n-
                        3*r*sin(2*i*pi/180)),3);
    nosound;
end;
```

Obsérvese que se pueden crear y emplear procedimientos dentro de procedimientos.

sound(50+i): instrucción que produce un sonido cuya frecuencia en hertz está dado por el valor entre paréntesis. El sonido se acaba cuando aparece la instrucción nosound.

for i:=0 to o div 2 do giro: esta proposición sirve para indicar que una cierta proposición debe repetirse un número predeterminado de veces. Para este caso i es un contador de tipo escalar e irá tomando valores desde 0 incrementándose hasta el valor de o div 2. Mientras estemos en el rango de valores de i se ejecuta la instrucción giro. En caso contrario termina el ciclo y continúa con la siguiente instrucción. Si cambiamos la instrucción to por downto, el contador irá disminuyendo de valor.

```
Procedure movimiento;
    Label 1;
    var x,y,ang:real;
    begin
        setviewport(0,0,getmaxx,getmaxy,true);
        t:=0;
        1: repeat
                begin
                    Vx:=Vo*Cos(Ango*Pi/180);
                    Vy:=Vo*Sin(Ango*Pi/180)-g*t;
                    V :=sqrt(sqr(Vx)+sqr(Vy));
                    X :=Vx*t+Xo;
```

```
                Y :=-1/2*g*sqr(t)+Vo*Sin(Ango*Pi/180)*t+Yo;
                if Vx<>0 then Ang:=arctan(Vy/Vx)*180/Pi
                        else Ang:=90;
                bola(6*r+trunc(x*5*r),trunc(y*5*r)+r);
                sound(400+round(20*g*y));
                 t:=t+incre
              end
        until  keypressed or (y<0)=true;
    if y>0 then
        begin
          if (((5*r*y)>100)=true and (getmaxy-trunc(5*r*y)>0) and
                  (trunc(5*r*x)-6*r<getmaxx)) then calcular(t);
          setcolor(3);
          circle(6*r+trunc(x*r*5),getmaxy-trunc(y*5*r)-r,r div 2);
          setfillstyle(1,1);
          floodfill(6*r+trunc(x*r*5),getmaxy-trunc(y*5*r)-r,3);
          setlinestyle(0,0,3);
          line(6*r+trunc(x*5*r),getmaxy-r-
                  trunc(y*5*r),6*r+trunc(x*r*5), getmaxy-r-
                  trunc(y*5*r+Vy*r/2));
          line(6*r+trunc(x*5*r),  getmaxy-r-trunc(y*5*r),
                  6*r+trunc(x*r*5+Vx*r/2),
                  getmaxy-r-trunc(y*5*r));
          line(6*r+trunc(x*5*r),getmaxy-r-trunc(y*5*r),
                  6*r+trunc(x*r*5+Vx*r/2),
                  getmaxy-r-trunc(y*5*r+Vy*r/2));
           setlinestyle(0,0,0);
          nosound;
           readln;readln;
          setfillstyle(0,0);
          floodfill(6*r+trunc(x*5*r),getmaxy-r-trunc(y*5*r),3);
          goto 1
        end
    setcolor(3);
    circle(6*r+trunc(x*5*r),getmaxy-r-trunc(y*5*r),r div 2);
    setfillstyle(1,1);
    floodfill(6*r+trunc(x*5*r),getmaxy-r-trunc(y*5*r),3);
    sound(50);delay(50);nosound;
end;
```

Label 1: permite declarar un rótulo en el programa, en este caso se puede saltar incondicionalmente desde cualquier parte del programa hasta la posición numerada con 1: mediante la instrucción goto 1.

repeat begin_end until keypressed or (y<0)=true: esta estructura sirve para especificar la ejecución reiterada de cierta proposición hasta cuando una determinada condición (expresión booleana) sea verdadera (true).

keypressed: expresión booleana que indica TRUE cuando ha sido presionada cualquier tecla y FALSE en caso contrario.

Los demás procedimientos creados como menu, mover, hombre, piso, emplean las mismas instrucciones en un orden lógico acorde con el problema físico que se plantea, de manera que el programa principal se reduce a los siguientes pasos:

```
begin
  c:=0;esc:=0;   incre:=0.02;Xo:=0;Yo:=0;Vo:=10;
                 Ango:=70;g:=9.8;r:=15;tar:=0;mopa:=0;
  initgraph(tar,mopa,'');
  iniciar('Programa elaborado por', 'Luis Fernando Madrid Z', 'Do-
                 cente U. Nacional y Autónoma');
  setcolor(3);
  mover;
  menu
end.
```

Este programa analiza el movimiento parabólico en diferentes aspectos, con opción de variar las condiciones iniciales como la velocidad, el ángulo, la posición horizontal, la altura, el valor de la gravedad (por omisión toma el valor de 9.8). Además tiene la posibilidad de poder visualizar la trayectoria seguida por el objeto y una retícula graduada en metros para determinar la posición en cualquier instante. También se puede variar el tamaño de la imagen (por omisión el tamaño es cero). Puede observarse el movimiento continuo o detenerlo presionando ENTER, mostrando el estado en que se encuentra el cuerpo en un instante dado tal como tiempo, posición, velocidad y ángulo, y retornar el movimiento volviendo a presionar la tecla ENTER. Un ejemplo de lo que el programa nos presenta se muestra en la figura 3 (Ver pág. siguiente)

Se espera que este programa sea una motivación para que el programador diseñe nuevos y mejores programas con miras a hacer de la Física un curso ameno y de fácil comprensión.

**Bibliografía**

BUITRAGO ALFONSO, Germán. **Turbo Pascal.** Universidad Nacional de Colombia Sede Manizales. 1994.

HOYOS S. EFRAIN Alberto. **Programación gráfica en Turbo Pascal.** Editorial TEX. Colombia . Enero de 1991.

YABORSKI, B. M. **Manual de Física**. Editorial MIR. Moscú. 1977.

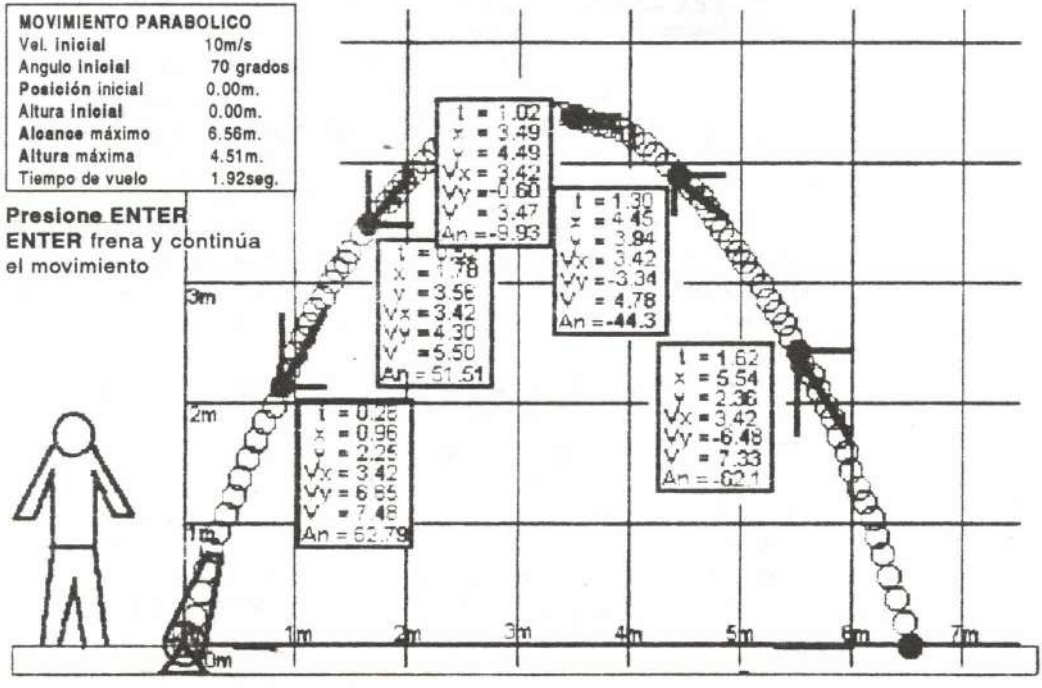


**Figura 3.** Análisis detallado del Movimiento parabólico