ANEXO 2: CÓDIGO DE PROGRAMACIÓN

frmMesures - 1

```
Option Explicit

'Declaración de las variables exógenas

Dim X(), Y(), R(), L(), lpol(), ve(), Lpoltot() As Double
Dim velo_recorrido, t_recorrido As Double
Dim n, elem As Long



Private Sub cmdRun_Click()
On Error GoTo ErrorHandler

'Variable dictionary

Dim objDocument As GeoMedia.Document
Dim objSelectedObjects As Pclient.DocSelectedObjects
Dim objObject As Object
Dim mobjConnection As Connection

Dim ObjPoints As PBasic.Points
Dim ObjPoint As PBasic.Point

Dim numelem As Long

'Get document selected objects

Set objDocument = gobjGeoApp.Document
Set objSelectedObjects = objDocument.SelectedObjects

'Asegurémonos de haber seleccionado un objeto

If objSelectedObjects.Count > 0 Then

    Set objObject = objSelectedObjects.Item(1)

    Dim objGSS As GeometryStorageService
    Dim objRecordset As GRecordset
    Dim objGeometry As PolylineGeometry

    Set objRecordset = objObject.Recordset

    objRecordset.MoveLast
    numelem = objRecordset.RecordCount
```

frmMesures - 2

```
    objRecordset.MoveFirst

    For elem = 1 To numelem

         If elem = 150 Then
            MsgBox " començo el 150"
        End If
         If elem = 275 Then
            MsgBox " començo el 275"
        End If

        If elem = 389 Then
            MsgBox " començo el 389"
        End If

        If elem = 500 Then
            MsgBox " començo el 500"
        End If

         If elem = 650 Then
            MsgBox " començo el 650"
        End If

         If elem = 800 Then
            MsgBox " començo el 800"
        End If


        'objRecordset.Bookmark = objObject.Bookmark

        'enviamos al objeto objGeometry el objeto seleccionado ndex para poderlo procesar

        Set objGSS = gobjGeoApp.CreateService("Geomedia.GeometryStorageService")
        Call objGSS.StorageToGeometry(objRecordset.GFields(objObject.GeometryFieldName).Value, objGeometry)
        Set objGSS = Nothing


        Set ObjPoints = objGeometry.Points

        Dim n1, n2, i, h As Long
        Dim j As Integer

        'número de puntos
        n1 = ObjPoints.Count
```

```
        'simplificamos el nº de puntos a uno de cada 2 - o el valor de txt.sim-

        n2 = Int((n1 - 2) / (TextSim.Text + 1))
        n = n2 + 2   'siguen existiendo n puntos!


        Dim cota(), a1(), a2(), a3(), a4(), a5(), p(), q() As Double
        ReDim X(n + 1), Y(n + 1), cota(n + 1), a1(n + 1), a2(n + 1), a3(n + 1), a4(n + 1), a5(n + 1), p(n + 1), q(n + 1), R(n + 1)
        ReDim A(n + 1), b(n + 1), c(n + 1), d(n + 1), e(n + 1), es(n + 1), fs(n + 1), gs(n + 1)

        Set ObjPoint = ObjPoints.Item(1)
            X(1) = ObjPoint.X
            Y(1) = ObjPoint.Y
            cota(1) = ObjPoint.z / 100 'Geomedia me da la cota en cm

        For i = 1 To n - 2
            Set ObjPoint = ObjPoints.Item((TextSim.Text + 1) * i + 1)
            X(i + 1) = ObjPoint.X
            Y(i + 1) = ObjPoint.Y
            cota(i + 1) = ObjPoint.z / 100 'Geomedia me da la cota en cm
        Next i

        Set ObjPoint = ObjPoints.Item(n1)
            X(n) = ObjPoint.X
            Y(n) = ObjPoint.Y
            cota(n) = ObjPoint.z / 100 'Geomedia me da la cota en cm

    'suavizamos el trazado tomando los puntos medios de los puntos medios

        If TxtSUA.Text >= 1 Then
            For j = 1 To TxtSUA.Text
                For i = 1 To n - 1
                    c(i) = (X(i) + X(i + 1)) / 2
                    d(i) = (Y(i) + Y(i + 1)) / 2
                Next i

                For i = 1 To n - 2
                    es(i) = (c(i) + c(i + 1)) / 2
                    fs(i) = (d(i) + d(i + 1)) / 2
                Next i

                For i = 1 To n - 2
                    X(i + 1) = es(i)
```

```
                    Y(i + 1) = fs(i)
                Next i
            Next j
        End If

    'calculamos ahora los puntos intermedios (una vez hecha la suavización)

        For i = 1 To n - 1
            A(i) = (X(i) + X(i + 1)) / 2
            b(i) = (Y(i) + Y(i + 1)) / 2
        Next i

    ' suavizamos también el relevo (cotas)

        If SuaRell.Text >= 1 Then
            For j = 1 To SuaRell.Text
                For i = 1 To n - 1
                    e(i) = (cota(i) + cota(i + 1)) / 2
                Next i

                For i = 1 To n - 2
                    gs(i) = (e(i) + e(i + 1)) / 2
                Next i

                For i = 1 To n - 2
                    cota(i + 1) = gs(i)
                Next i
            Next j
        End If

    If elem = 389 Then
            MsgBox "he agafat la geometria del 389"
        End If

    'Radio de curvatura i longitudes

        Dim t(), s(), alpha() As Double
        ReDim t(n + 1), s(n + 1), alpha(n + 1)
        Dim w(), z(), beta() As Double
        ReDim w(n + 1), z(n + 1), beta(n + 1)
        ReDim L(n + 1), lpol(n + 1)

    'compruebo que tengamos más de dos puntos
```

frmMesures - 5

```
        Dim Ltotal, Ltot() As Double
        Dim Lpoltotal  As Double
        ReDim Ltot(n + 1)
        ReDim Lpoltot(n + 1)

        If n < 3 Then

                R(1) = 51000
                L(1) = Math.Sqr((X(n) - X(1)) ^ 2 + (Y(n) - Y(1)) ^ 2)
                lpol(1) = L(1)
                Ltotal = lpol(1)
                Lpoltotal = L(1)
               ' ListLtotal.AddItem Format(Ltotal / 1000, "####0.000")
               ' ListLpoltotal.AddItem Format(Lpoltotal / 1000, "####0.000")

        Else

            For i = 1 To n - 2
            'Excepción: los puntos estan sobre una "horizontal"
            If Abs(Y(i + 1) - Y(i)) < 0.01 Then    'la precisión de les coordenades de los puntos es de 1 cm
                If Abs(Y(i + 2) - Y(i + 1)) < 0.01 Then
                    R(i) = 10000
                    L(i) = (1 / 2) * ((Math.Sqr((X(i + 1) - X(i)) ^ 2 + (Y(i + 1) - Y(i)) ^ 2) + Math.Sqr((X(i + 2) - X(i + 1)) ^ 2 +
(Y(i + 2) - Y(i + 1)) ^ 2)))
                    lpol(i) = L(i)

                Else
                    p(i) = (X(i) + X(i + 1)) / 2
                    q(i) = (1 / (2 * (Y(i + 2) - Y(i + 1)))) * ((X(i) + X(i + 1)) * (X(i + 1) - X(i + 2)) + X(i + 2) ^ 2 - X(i + 1) ^
2 + Y(i + 2) ^ 2 - Y(i + 1) ^ 2)
                    R(i) = Math.Sqr((X(i + 1) - p(i)) ^ 2 + (Y(i + 1) - q(i)) ^ 2)

                    t(i) = Abs((X(i + 1) - X(i)) / 2)
                    s(i) = ((A(i) - p(i)) ^ 2 + (b(i) - q(i)) ^ 2) ^ (1 / 2)
                    alpha(i) = Math.Atn(t(i) / s(i))

                    w(i) = (1 / 2) * (((X(i + 2) - X(i + 1)) ^ 2 + (Y(i + 2) - Y(i + 1)) ^ 2) ^ (1 / 2))
                    z(i) = ((A(i + 1) - p(i)) ^ 2 + (b(i + 1) - q(i)) ^ 2) ^ (1 / 2)
                    beta(i) = Math.Atn(w(i) / z(i))

                    L(i) = R(i) * (alpha(i) + beta(i))
                    lpol(i) = (1 / 2) * ((Math.Sqr((X(i + 1) - X(i)) ^ 2 + (Y(i + 1) - Y(i)) ^ 2) + Math.Sqr((X(i + 2) - X(i + 1)) ^ 2
 + (Y(i + 2) - Y(i + 1)) ^ 2)))
                End If
```

```
            Else


                If Abs(X(i + 1) - X(i)) < 0.01 Then 'Excepción: los puntos estan sobre una "vertical"

                    If Abs(X(i + 2) - X(i + 1)) < 0.01 Then
                        R(i) = 10000
                        L(i) = (1 / 2) * ((Math.Sqr((X(i + 1) - X(i)) ^ 2 + (Y(i + 1) - Y(i)) ^ 2) + Math.Sqr((X(i + 2) - X(i + 1)) ^
2 + (Y(i + 2) - Y(i + 1)) ^ 2)))
                        lpol(i) = L(i)
                    Else
                        q(i) = (Y(i) + Y(i + 1)) / 2
                        p(i) = (1 / (X(i + 1) - X(i + 2))) * (q(i) * (Y(i + 2) - Y(i + 1)) - ((Y(i + 2) ^ 2 - Y(i + 1) ^ 2) - (X(i + 1
) ^ 2 - X(i + 2) ^ 2)) / 2)
                        R(i) = Math.Sqr((X(i + 1) - p(i)) ^ 2 + (Y(i + 1) - q(i)) ^ 2)

                        t(i) = Abs((1 / 2) * (Y(i + 1) - Y(i)))
                        s(i) = ((A(i) - p(i)) ^ 2 + (b(i) - q(i)) ^ 2) ^ (1 / 2)
                        alpha(i) = Math.Atn(t(i) / s(i))

                        w(i) = (1 / 2) * (((X(i + 2) - X(i + 1)) ^ 2 + (Y(i + 2) - Y(i + 1)) ^ 2) ^ (1 / 2))
                        z(i) = ((A(i + 1) - p(i)) ^ 2 + (b(i + 1) - q(i)) ^ 2) ^ (1 / 2)
                        beta(i) = Math.Atn(w(i) / z(i))

                        L(i) = R(i) * (alpha(i) + beta(i))
                        lpol(i) = (1 / 2) * ((Math.Sqr((X(i + 1) - X(i)) ^ 2 + (Y(i + 1) - Y(i)) ^ 2) + Math.Sqr((X(i + 2) - X(i + 1))
 ^ 2 + (Y(i + 2) - Y(i + 1)) ^ 2)))
                    End If
                Else

                    If Abs(X(i + 2) - X(i + 1)) < 0.01 Then
                        q(i) = (Y(i + 1) + Y(i + 2)) / 2
                        p(i) = (1 / (X(i) - X(i + 1))) * (q(i) * (Y(i + 1) - Y(i)) - ((Y(i + 1) ^ 2 - Y(i) ^ 2) - (X(i) ^ 2 - X(i + 1)
 ^ 2)) / 2)
                        R(i) = Math.Sqr((X(i + 1) - p(i)) ^ 2 + (Y(i + 1) - q(i)) ^ 2)

                        t(i) = Abs((1 / 2) * (Y(i + 1) - Y(i)))
                        s(i) = ((A(i) - p(i)) ^ 2 + (b(i) - q(i)) ^ 2) ^ (1 / 2)

                        alpha(i) = Math.Atn(t(i) / s(i))
                        w(i) = (1 / 2) * (((X(i + 2) - X(i + 1)) ^ 2 + (Y(i + 2) - Y(i + 1)) ^ 2) ^ (1 / 2))
                        z(i) = ((A(i + 1) - p(i)) ^ 2 + (b(i + 1) - q(i)) ^ 2) ^ (1 / 2)
                        beta(i) = Math.Atn(w(i) / z(i))
```

```
                        L(i) = R(i) * (alpha(i) + beta(i))
                        lpol(i) = (1 / 2) * ((Math.Sqr((X(i + 1) - X(i)) ^ 2 + (Y(i + 1) - Y(i)) ^ 2) + Math.Sqr((X(i + 2) - X(i + 1))
 ^ 2 + (Y(i + 2) - Y(i + 1)) ^ 2)))

                    Else
                        'Excepción: los puntos estan sobre una recta "oblicua"
                        If Abs(((Y(i + 2) - Y(i + 1)) / (X(i + 2) - X(i + 1))) - ((Y(i + 1) - Y(i)) / (X(i + 1) - X(i)))) < 0.001 Then
                            R(i) = 10000
                            L(i) = (1 / 2) * ((Math.Sqr((X(i + 1) - X(i)) ^ 2 + (Y(i + 1) - Y(i)) ^ 2) + Math.Sqr((X(i + 2) - X(i + 1)
) ^ 2 + (Y(i + 2) - Y(i + 1)) ^ 2)))
                            lpol(i) = L(i)

                        Else

                    'Caso genérico

                            a1(i) = ((Y(i + 2) ^ 2) - (Y(i + 1) ^ 2)) - ((X(i + 1) ^ 2) - (X(i + 2) ^ 2))
                            a2(i) = (Y(i + 2) - Y(i + 1)) / (Y(i + 1) - Y(i))
                            a3(i) = ((Y(i + 1) ^ 2) - (Y(i) ^ 2)) + ((X(i + 1) ^ 2) - (X(i) ^ 2))
                            a4(i) = ((Y(i + 2) - Y(i + 1)) * (X(i) - X(i + 1))) / (Y(i + 1) - Y(i))
                            a5(i) = X(i + 2) - X(i + 1)
                            p(i) = (a1(i) - (a2(i) * a3(i))) / (2 * (a4(i) + a5(i)))
                            q(i) = (((X(i) - X(i + 1)) * p(i)) / (Y(i + 1) - Y(i))) + (((Y(i + 1) ^ 2 - Y(i) ^ 2) + (X(i + 1) ^ 2 - X(
i) ^ 2)) / (2 * (Y(i + 1) - Y(i))))

                            R(i) = Math.Sqr((X(i + 1) - p(i)) ^ 2 + (Y(i + 1) - q(i)) ^ 2)

                            t(i) = 1 / 2 * Math.Sqr((X(i + 1) - X(i)) ^ 2 + (Y(i + 1) - Y(i)) ^ 2)
                            s(i) = Math.Sqr((A(i) - p(i)) ^ 2 + (b(i) - q(i)) ^ 2)
                            alpha(i) = Math.Atn(t(i) / s(i))

                            w(i) = 1 / 2 * Math.Sqr((X(i + 2) - X(i + 1)) ^ 2 + (Y(i + 2) - Y(i + 1)) ^ 2)
                            z(i) = Math.Sqr((A(i + 1) - p(i)) ^ 2 + (b(i + 1) - q(i)) ^ 2)
                            beta(i) = Math.Atn(w(i) / z(i))

                            L(i) = R(i) * (alpha(i) + beta(i))
                            lpol(i) = 1 / 2 * (Math.Sqr((X(i + 1) - X(i)) ^ 2 + (Y(i + 1) - Y(i)) ^ 2) + Math.Sqr((X(i + 2) - X(i + 1)
) ^ 2 + (Y(i + 2) - Y(i + 1)) ^ 2))

                        End If
                    End If
                End If
            End If
```

```
            Next i

      'Longitud total del tramo

        L(1) = L(1) + alpha(1) * R(1)
        L(n - 2) = L(n - 2) + beta(n - 2) * R(n - 2)

    '    L(1) = L(1) + (1 / 2) * (Math.Sqr((X(2) - X(1)) ^ 2 + (Y(2) - Y(1)) ^ 2))
    '    L(n - 2) = L(n - 2) + (1 / 2) * (Math.Sqr((X(n) - X(n - 1)) ^ 2 + (Y(n) - Y(n - 1)) ^ 2))

        Ltot(1) = L(1)
        For i = 2 To n - 2
             Ltot(i) = L(i) + Ltot(i - 1)
        Next i

        Ltotal = Ltot(n - 2)

      'Longitud total de la polilínea

        lpol(1) = lpol(1) + (1 / 2) * (Math.Sqr((X(2) - X(1)) ^ 2 + (Y(2) - Y(1)) ^ 2))
        lpol(n - 2) = lpol(n - 2) + (1 / 2) * (Math.Sqr((X(n) - X(n - 1)) ^ 2 + (Y(n) - Y(n - 1)) ^ 2))

        Lpoltot(1) = lpol(1)
        For i = 2 To n - 2
            Lpoltot(i) = lpol(i) + Lpoltot(i - 1)
        Next i

        Lpoltotal = Lpoltot(n - 2)

'        For i = 1 To n - 2
'            ListR.AddItem Format(R(i), "####0")
'            ListLpol.AddItem Format(lpol(i), "####0.0")
'        Next i
'
'             ListLtotal.AddItem Format(Ltotal / 1000, "####0.000")
'             ListLpoltotal.AddItem Format(Lpoltotal / 1000, "####0.000")

    End If

    'Velocidad específica en función del radio

        Dim Vplanej As Double, Vpolplanej As Double
        ReDim ve(n + 1)
        ReDim vpola(n + 1)
```

```
        If n < 3 Then
            ve(1) = 90
            Vpolplanej = 90
'            ListVpolplanej.AddItem Format(Vpolplanej, "####0.0")


        Else

        'Ferrocarril

            If optTgv.Value = True Then
                For i = 1 To n - 2
                    ve(i) = 3.6 * (R(i) * (Asc.Text + 9.8 * (Peralt.Text / DCarrils.Text))) ^ (1 / 2)
                    If ve(i) > 350 Then
                        ve(i) = 350
                    End If
'                    ListVe.AddItem Format(ve(i), "####0.0")
                Next i
            End If

        'Carretera

            Dim Per() As Double
            ReDim Per(n + 1)

            If optCarr1.Value = True Then     'carretera del primer grupo: autopistas, autovías, vías rápidas i carreteras C-100

                    For i = 1 To n - 2
                        Select Case R(i)         'la siguiente expresión se obtiene de la Norma de Trazado
                            Case Is < 700
                                Per(i) = 8 / 100
                            Case 700 To 5000
                                Per(i) = (8 - 7.3 * ((1 - (700 / R(i))) ^ 1.3)) / 100
                            Case 5000 To 7500
                                Per(i) = 2 / 100
                            Case Is > 7500
                                Per(i) = 0
                        End Select

                        ve(i) = 0.5 * (-0.1125 * R(i) + Math.Sqr(0.01267 * (R(i) ^ 2) + 508 * R(i) * (0.193 + Per(i))))
                        If ve(i) > 150 Then       'la velocidad específica máxima fijada por la Norma en carreteras del primer grupo
```

```
                            ve(i) = 150             'és de 150 km/h (ya la 'decaparemos' luego con vmax)
                        End If
'                        ListVe.AddItem Format(ve(i), "####0.0")
                    Next i

            End If


            If optCarr2.Value = True Then     'carretera del segundo grupo (carreteras C-80, C-60 i C-40)

                        For i = 1 To n - 2
                            Select Case R(i)
                                Case Is < 350
                                    Per(i) = 7 / 100
                                Case 350 To 2500
                                    Per(i) = (7 - 6.08 * ((1 - (350 / R(i))) ^ 1.3)) / 100
                                Case 2500 To 3500
                                    Per(i) = 2 / 100
                                Case Is > 3500
                                    Per(i) = 0
                            End Select

                            If R(i) < 250 Then
                                ve(i) = 0.5 * (-0.18415 * R(i) + Math.Sqr(0.03391 * (R(i) ^ 2) + 508 * R(i) * (0.238 + Per(i))))
                            Else
                                ve(i) = 0.5 * (-0.1125 * R(i) + Math.Sqr(0.01267 * (R(i) ^ 2) + 508 * R(i) * (0.193 + Per(i))))
                            End If

                            If ve(i) > 110 Then      'la velocidad específica máxima fijada por la Norma en carreteras del segundo gru
po
                                ve(i) = 110             'es de 110 km/h
                            End If
'                            ListVe.AddItem Format(ve(i), "####0.0")
                        Next i

            End If


        'Velocidad de planeamiento: media harmónica de las velocidades específicas de cada tramo

            Dim Mtotal
            Dim m(), Mtot() As Double
            ReDim m(n + 1), Mtot(n + 1)
```

```
            For i = 1 To n - 2
                m(i) = L(i) / ve(i)
            Next i

            Mtot(1) = m(1)
            For i = 2 To n - 2
                Mtot(i) = m(i) + Mtot(i - 1)
            Next i

            Mtotal = Mtot(n - 2)

            Vplanej = Ltotal / Mtotal

            'velo planej con lpol (poligional)

            Dim Mpoltotal
            Dim mpol(), Mpoltot() As Double
            ReDim mpol(n + 1), Mpoltot(n + 1)

            For i = 1 To n - 2
                mpol(i) = lpol(i) / ve(i)
            Next i

            Mpoltot(1) = mpol(1)
            For i = 2 To n - 2
                Mpoltot(i) = mpol(i) + Mpoltot(i - 1)
            Next i

            Mpoltotal = Mpoltot(n - 2)

            Vpolplanej = Lpoltotal / Mpoltotal

'            ListVpolplanej.AddItem Format(Vpolplanej, "####0.0")


            'UNA VEZ CALCULADOS LOS RADIOS, TENEMOS LOS VECTORES DE LONGITUD Y VELOCIDAD ESPECÍFICA
            'PASAMOS A CALCULAR EL RECORRIDO DE VELOCIDADES Y LA VELOCIDAD DE RECORRIDO

            'Dimensionamento de variables
            Dim lim, ac As Long
            Dim v0, vf, decel, v1, v2, Fv1, Fv2, A_ctt, D_ctt, Massa, VelMax, Interval, FA_ctt, FA_vel, FF, FF_c, acc_max, L_total As
Double
            Dim pendent(), va(), vac(), vfr(), vfre(), vini(), vfin(), espai(), acc(), x_acum(), velo(), v_real(), x_real(), t_real(),
 x1(), x2(), t1(), t2(), temps(), vel_x1(), x_accel(), xa(), xf(), xctt(), ta(), tf(), tctt(), t_total(), v_mitjana(), vel_fre(), x_re
```

```
c(), refer(), t_rec(), v_pic(), x_pic(), t_pic(), x_dib(), v_dib(), t_dib(), limit() As Double
            Dim subtram() As Long

            'Consideramos valores

            'Factores que afectan a la Fuerza Aerodinámica (valores nulos implican no considerarla)
            FA_ctt = CDbl(frmMesures.FA_ctt.Text) 'Factor constante de oposición al esfuerzo tractor
            FA_vel = CDbl(frmMesures.FA_vel.Text) 'Factor multiplicador de v^2

            'Factor de fricción de la rueda(el valor FF=1 implica no considerarla)
            FF = CDbl(frmMesures.FF.Text)
            FF_c = 10 * FF / 9.8

            'Deceleración
            decel = CDbl(frmMesures.Dec.Text)

            'Aceleración máxima
            acc_max = CDbl(frmMesures.AccMax.Text)

            'Masa
            Massa = CDbl(frmMesures.Massa.Text) 'kg
            A_ctt = 1000 / (Massa * 9.8)

            'Velocidades inicial y final
            v0 = CDbl(frmMesures.VeloIni.Text) / 3.6 'se considera en km/h y se convierte a m/s
            vf = CDbl(frmMesures.VeloFin.Text) / 3.6 'se considera en km/h y se convierte a m/s

            'Esfuerzos tractores y velocidades correspondientes
            Fv1 = CDbl(frmMesures.F1.Text) * 1000 'se considera en KN y se convierte a N
            Fv2 = CDbl(frmMesures.F2.Text) * 1000 'se considera en KN y se convierte a N
            v1 = CDbl(frmMesures.v1.Text) / 3.6 'se considera en km/h y se convierte a m/s
            v2 = CDbl(frmMesures.v2.Text) / 3.6 'se considera en km/h y se convierte a m/s
            D_ctt = (Fv1 - Fv2) / (v2 - v1) 'en N·s/m

            'Intervalo de velocidades para el cálculo de la aceleración
            'A intervalo más pequeño, más precisión
            Interval = CDbl(frmMesures.Interval.Text) / 3.6 'se considera en km/h y se convierte a m/s

            'Velocidad máxima admisible
            VelMax = CDbl(frmMesures.Vmax.Text) / 3.6 'se considera en km/h y se convierte a m/s
            lim = (VelMax - v1) / Interval

            'Dimensionamos los vectores y las matrices en función de n y lim
            ReDim pendent(n + 1), va(n + 1), vac(n + 1), vfr(n + 1), vfre(n + 1), vini(n + 1), vfin(n + 1), x_acum(n + 1), fin(n + 1),
```

```
 espai(n + 1, lim + 1), acc(n + 1, lim + 1), velo(n + 1, lim + 1), v_real(n + 1, lim + 1), x_real(n + 1, lim + 1), t_real(n + 1, lim +
 1), x1(n + 1), x2(n + 1), t1(n + 1), t2(n + 1), temps(n + 1, lim + 1), vel_x1(n + 1, lim + 1), x_accel(n + 1, lim + 1), xa(n + 1), xf
(n + 1), xctt(n + 1), ta(n + 1), tf(n + 1), tctt(n + 1), t_total(n + 1), v_mitjana(n + 1), vel_fre(n + 1, lim + 1), x_rec(n + 1, lim +
 1), refer(n + 1), t_rec(n + 1, lim + 1), v_pic(n + 1), x_pic(n + 1), t_pic(n + 1), x_dib((n + 1) * (lim + 1)), v_dib((n + 1) * (lim +
 1)), x_dib((n + 1) * (lim + 1)), t_dib((n + 1) * (lim + 1)), limit(n + 1) As Double
            ReDim subtram(n + 1) As Long


            'Obtenemos la pendiente de los datos gráficos

            For i = 1 To n - 2

                pendent(i) = 1 / 2 * 1000 * (cota(i + 2) - cota(i)) / lpol(i) 'en mm/m


            'Limitamos la pendiente según el tipo de vía

                If optTgv.Value = True Then
                    If pendent(i) > 20 Then
                        pendent(i) = 20 'máx pendiente en mm/m
                    End If
                    If pendent(i) < -20 Then
                        pendent(i) = -20
                    End If
                End If

                If optCarr1.Value = True Then
                    If pendent(i) > 50 Then
                        pendent(i) = 50 'máx pendiente en carreteras del tipo 1 (5%)
                    End If
                    If pendent(i) < -50 Then
                        pendent(i) = -50
                    End If
                End If

                If optCarr2.Value = True Then
                    If pendent(i) > 100 Then
                        pendent(i) = 100 'máx pendiente en carreteras del tipo 2 (10%)
                    End If
                    If pendent(i) < -100 Then
                        pendent(i) = -100
                    End If
                End If
```

```
            Next i

            'Limitamos la velocidad específica con VelMáx de la vía!

            For i = 1 To n2

                ve(i) = ve(i) / 3.6 'se convierte a m/s

                If ve(i) > VelMax Then
                    ve(i) = VelMax
                End If

            Next i


            'CÁLCULO DE VELOCIDADES ACELERANDO Y FRENANDO
            vac(0) = v0
            vfre(n2) = vf

            'Cálculo de velocidad frenando
            For i = n2 - 1 To 0 Step -1
                vfr(i) = Math.Sqr(vfre(i + 1) ^ 2 - 2 * decel * lpol(i + 1))
                If vfr(i) < ve(i + 1) Then
                    vfre(i) = vfr(i)
                    Else: vfre(i) = ve(i + 1)
                End If
            Next i

            'Cálculo de velocidad acelerando

            For i = 1 To n2

                'primer tramo (hasta v1)
            '* (a partir de ahora siempre que encontremos el segundo subíndice de las matrices igual a 0 querrá decir que hablamos de
v <= v1) *

                If vac(i - 1) < v1 Then
                    acc(i, 0) = accel(0, 0.75 * v1, pendent(i), A_ctt, Fv1, v1, FA_ctt, FA_vel, FF_c)
                    If acc(i, 0) > acc_max Then
                        acc(i, 0) = acc_max
                    End If

                    'Calculamos la velocidad que considero acelerando con acc(i,0) durante todo el tramo (lpol(i))
                    va(i) = Math.Sqr(vac(i - 1) ^ 2 + 2 * acc(i, 0) * lpol(i))
```

```
                    If va(i) > v1 Then
                        'si esta velocidad es más grande que v1 sólo aceleramos con acc(i,0) hasta v1 y tenemos que pasar al segundo t
ramo
                        espai(i, 0) = (v1 ^ 2 - vac(i - 1) ^ 2) / (2 * acc(i, 0))
                        velo(i, 0) = v1
                        va(i) = 0
                    Else
                        'en caso contrario hemos hallado va(i)
                        espai(i, 0) = lpol(i)
                        va(i) = Math.Sqr(vac(i - 1) ^ 2 + 2 * acc(i, 0) * lpol(i))
                    End If

                Else
                    'no existe el primer tramo de velocidades
                    espai(i, 0) = 0
                    velo(i, 0) = vac(i - 1)
                End If

                'definimos el vector que irá recogiendo el espacio acumulado
                x_acum(i) = espai(i, 0)

                'segundo tramo (para v > v1)
                If va(i) = 0 Then
                    For j = 1 To lim
                        'en este segundo tramo de velocidades la aceleración se define para cada subtramo, siendo 'lim' el nº máx de s
ubtramos posibles
                        acc(i, j) = accel(D_ctt, velo(i, 0) + (j - 1 / 2) * Interval, pendent(i), A_ctt, Fv1, v1, FA_ctt, FA_vel, FF_c
)
                        If acc(i, j) > acc_max Then
                            acc(i, 0) = acc_max
                        End If

                        If acc(i, j) < 0.0001 Then
                            acc(i, j) = 0.0001
                        End If

                        espai(i, j) = ((velo(i, 0) + j * Interval) ^ 2 - (velo(i, 0) + (j - 1) * Interval) ^ 2) / (2 * acc(i, j))
                        velo(i, j) = velo(i, 0) + j * Interval
                        x_acum(i) = x_acum(i) + espai(i, j)
                        If x_acum(i) > lpol(i) Then Exit For
                    Next j

                    va(i) = Math.Sqr((velo(i, j)) ^ 2 - 2 * acc(i, j) * (x_acum(i) - lpol(i)))
```

```
                End If

                'Limitamos la velocidad acelerando a la velocidad específica del tramo

                If va(i) < ve(i) Then
                    vac(i) = va(i)
                Else
                    vac(i) = ve(i)
                End If

            Next i

            'CÁLCULO DE VELOCIDADES INICIALES Y FINALES DE CADA TRAMO

            For i = 1 To n2

                If vac(i - 1) < vfre(i - 1) Then
                    vini(i) = vac(i - 1)
                Else
                    vini(i) = vfre(i - 1)
                End If


                If vac(i) < vfre(i) Then
                    vfin(i) = vac(i)
                Else
                    vfin(i) = vfre(i)
                End If

            Next i

            '-------------------------------------------------------------------------------------------------------------------
---------------

            'CÁLCULO DE x1, x2

              'Es necesario volver a recorrer el tramo(ahora conociendo v.inicial y final de cada tramo)

            For i = 1 To n2

              'Reiniciamos variables:
                fin(i) = 0 '(Variable de control, fin(i) = 1 quiere decir que hemos llegado a x1)
                vel_x1(i, 0) = vini(i) '(matriz que recogerá las velocidades alcanzadas en cada subtramo)
                x1(i) = 0 'vector acumulativo de x
```

```
    t1(i) = 0 'vector acumulativo de t
    subtram(i) = 0 '(guarda el subtramo en el que acaba el bucle)
    x_rec(i, 0) = 0 '(espacio recorrido en cada subtramo)
    acc(i, 0) = 0

  'primer tramo (hasta v1)

    If vini(i) < v1 Then
        acc(i, 0) = accel(0, 0.75 * v1, pendent(i), A_ctt, Fv1, v1, FA_ctt, FA_vel, FF_c)
        If acc(i, 0) > acc_max Then
            acc(i, 0) = acc_max
        End If

        If ve(i) <= v1 Then
            x_accel(i, 0) = (ve(i) ^ 2 - vini(i) ^ 2) / (2 * acc(i, 0))
            x_rec(i, 0) = x_accel(i, 0)
            x1(i) = x_rec(i, 0)
            temps(i, 0) = (ve(i) - vini(i)) / acc(i, 0)
            t_rec(i, 0) = temps(i, 0)
            t1(i) = t_rec(i, 0)
            vel_x1(i, 0) = ve(i)
            fin(i) = 1
        Else
            x_accel(i, 0) = (v1 ^ 2 - vini(i) ^ 2) / (2 * acc(i, 0))
            x_rec(i, 0) = x_accel(i, 0)
            x1(i) = x_accel(i, 0)
            temps(i, 0) = (v1 - vini(i)) / acc(i, 0)
            t1(i) = temps(i, 0)
            t_rec(i, 0) = temps(i, 0)
            vel_x1(i, 0) = v1
            fin(i) = 0
        End If

    Else
      '(esto no es necesario, es para hacer más "inteligible" el codi)
        fin(i) = 0
        vel_x1(i, 0) = vini(i)
        x1(i) = 0
        t1(i) = 0
        subtram(i) = 0
        x_rec(i, 0) = 0
        acc(i, 0) = 0

    End If
```

```
                'segundo tramo (pera v > v1)
                If fin(i) = 0 Then

                    For j = 1 To lim
                        acc(i, j) = accel(D_ctt, vel_x1(i, 0) + (j - 1 / 2) * Interval, pendent(i), A_ctt, Fv1, v1, FA_ctt, FA_vel, FF
_c)
                        If acc(i, j) > acc_max Then
                            acc(i, j) = acc_max
                        End If

                        If acc(i, j) < 0.0001 Then
                            acc(i, j) = 0.0001
                        End If

                        vel_x1(i, j) = vel_x1(i, 0) + j * Interval
                        If vel_x1(i, j) >= ve(i) Then
                            vel_x1(i, j) = ve(i)
                        End If
                        x_accel(i, j) = (vel_x1(i, j) ^ 2 - vel_x1(i, j - 1) ^ 2) / (2 * acc(i, j))
                        temps(i, j) = (vel_x1(i, j) - vel_x1(i, j - 1)) / acc(i, j)
                        x_rec(i, j) = x_rec(i, j - 1) + x_accel(i, j)
                        t_rec(i, j) = t_rec(i, j - 1) + temps(i, j)
                        x1(i) = x1(i) + x_accel(i, j)
                        t1(i) = t1(i) + temps(i, j)
                        subtram(i) = j
                        If vel_x1(i, j) = ve(i) Then Exit For
                    Next j

                End If

                'Cálculo de x2(i)

                x2(i) = (vfin(i) ^ 2 - ve(i) ^ 2) / (2 * decel)
                t2(i) = (vfin(i) - ve(i)) / decel

            '-----------------------------------------------------------------------------------------------------------------
---

                'DIFERENCIANDO CASOS

                'CASO "FÁCIL" (en carreteras era el CASO 2!)

                If x1(i) + x2(i) < lpol(i) Then
```

```
                    xa(i) = x1(i) 'espacio acelerando
                    ta(i) = t1(i) 'tiempo acelerando
                    xf(i) = x2(i) 'espacio frenando
                    tf(i) = t2(i) 'tiempo frenando
                    xctt(i) = lpol(i) - (xa(i) + xf(i)) 'espacio a velocidad constante
                    tctt(i) = xctt(i) / ve(i) 'tiempo a velocidad constante



                    'Puntos de (espacio, velocidad, tiempo) reales del tramo

                    For j = 0 To subtram(i)
                        x_real(i, j) = x_accel(i, j)
                        v_real(i, j) = vel_x1(i, j)
                        t_real(i, j) = temps(i, j)

                    Next j

                    x_real(i, subtram(i) + 1) = xctt(i)
                    v_real(i, subtram(i) + 1) = ve(i)
                    t_real(i, subtram(i) + 1) = tctt(i)

                    x_real(i, subtram(i) + 2) = xf(i)
                    v_real(i, subtram(i) + 2) = vfin(i)
                    t_real(i, subtram(i) + 2) = tf(i)

                    limit(i) = subtram(i) + 3

                Else

                    'CASO COMPLICADO (en carreteras era el CASO 1)

                    ta(i) = 0 'Estos pueden o no ser 0 al final del 'bucle'
                    xa(i) = 0
                    tf(i) = 0
                    xf(i) = 0
                    refer(i) = 0

                    tctt(i) = 0 'Estos son siempre 0 en este caso
                    xctt(i) = 0

                    'Caso particular (puede dar problemas si no lo tratamos a parte)

                    If vini(i) = vfre(i - 1) And vfin(i) = vfre(i) Then
```

```
                    'Frenamos durante todo el tramo
                     xf(i) = lpol(i)
                     tf(i) = (vfin(i) - vini(i)) / decel

                     'Puntos de (espacio, velocidad) reales del tramo
                     x_real(i, 0) = lpol(i)
                     v_real(i, 0) = vfin(i)
                     t_real(i, 0) = tf(i)

                     limit(i) = 1

                Else

                    For h = 0 To subtram(i)
                        If x_rec(i, h) >= lpol(i) Then
                            If h = 0 Then
                                vel_fre(i, 0) = vfre(i)
                                vel_x1(i, 0) = vac(i)
                                x_rec(i, 0) = lpol(i)
                                x_accel(i, 0) = lpol(i)
                                temps(i, 0) = (vac(i) - vini(i)) / acc(i, 0)
                            Else
                                vel_fre(i, h) = vfre(i)
                                vel_x1(i, h) = vac(i)
                                x_rec(i, h) = lpol(i)
                                x_accel(i, h) = lpol(i) - x_rec(i, h - 1)
                                temps(i, h) = (vel_x1(i, h) - vel_x1(i, h - 1)) / acc(i, h)
                            End If
                        Else
                            vel_fre(i, h) = Math.Sqr(vfre(i) ^ 2 - 2 * decel * (lpol(i) - x_rec(i, h)))
                        End If

                        refer(i) = h

                        If vel_fre(i, h) >= vel_x1(i, h) Then
                            ta(i) = ta(i) + temps(i, h)
                            xa(i) = x_rec(i, h)
                            If xa(i) >= lpol(i) Then
                               'Aceleramos durante todo el tramo
                                xa(i) = lpol(i)
                                ta(i) = ta(i) - (vel_x1(i, h) - vfin(i)) / acc(i, h)
                                tf(i) = 0
                                xf(i) = 0
```

```
                                'Puntos de (espacio, velocidad) reales del tramo
                                If refer(i) = 0 Then
                                    x_real(i, refer(i)) = lpol(i)
                                    v_real(i, refer(i)) = vfin(i)
                                    t_real(i, refer(i)) = ta(i)

                                Else
                                    For j = 0 To refer(i) - 1
                                        x_real(i, j) = x_accel(i, j)
                                        v_real(i, j) = vel_x1(i, j)
                                        t_real(i, j) = temps(i, j)
                                    Next j

                                    x_real(i, refer(i)) = lpol(i) - x_rec(i, refer(i) - 1)
                                    v_real(i, refer(i)) = vfin(i)
                                    t_real(i, refer(i)) = ta(i) - t_rec(i, refer(i) - 1)
                                End If

                                limit(i) = refer(i) + 1

                                GoTo 10

                            End If
                        Else
                            Exit For

                        End If

                    Next h

                   'Aquí aprovechamos las fórmulas de carreteras para afinar el cálculo (si no lo hacemos hay casos que pueden apa
recer errores)

                    If refer(i) = 0 Then

                        'El cruce se da antes de v1
                        x_pic(i) = (vel_fre(i, 0) ^ 2 - vini(i) ^ 2 - 2 * decel * x_accel(i, 0)) / (2 * acc(i, 0) - 2 * decel)
                        xa(i) = x_pic(i)
                        v_pic(i) = Math.Sqr(vini(i) ^ 2 + 2 * acc(i, 0) * x_pic(i))
                        t_pic(i) = (v_pic(i) - vini(i)) / acc(i, refer(i))
                        ta(i) = t_pic(i)
                        xf(i) = lpol(i) - xa(i)
                        tf(i) = (vfin(i) - v_pic(i)) / decel
```

```
                            'Puntos de (espacio, velocidad) reales del tramo
                            x_real(i, 0) = xa(i)
                            v_real(i, 0) = v_pic(i)
                            t_real(i, 0) = t_pic(i)

                            x_real(i, 1) = xf(i)
                            v_real(i, 1) = vfin(i)
                            t_real(i, 1) = tf(i)

                            limit(i) = 2

                        Else
                            x_pic(i) = (vel_fre(i, refer(i)) ^ 2 - vel_x1(i, refer(i) - 1) ^ 2 - 2 * decel * x_accel(i, refer(i))) / (
2 * acc(i, refer(i)) - 2 * decel)
                            xa(i) = xa(i) + x_pic(i)
                            v_pic(i) = Math.Sqr(vel_x1(i, refer(i) - 1) ^ 2 + 2 * acc(i, refer(i)) * x_pic(i))
                            t_pic(i) = (v_pic(i) - vel_x1(i, refer(i) - 1)) / acc(i, refer(i))
                            ta(i) = ta(i) + t_pic(i)
                            xf(i) = lpol(i) - xa(i)
                            tf(i) = (vfin(i) - v_pic(i)) / decel

                            'Puntos de (espacio, velocidad, tiempo) reales del tramo
                            For j = 0 To refer(i) - 1
                                x_real(i, j) = x_accel(i, j)
                                v_real(i, j) = vel_x1(i, j)
                                t_real(i, j) = temps(i, j)
                            Next j

                            x_real(i, refer(i)) = x_pic(i)
                            v_real(i, refer(i)) = v_pic(i)
                            t_real(i, refer(i)) = t_pic(i)

                            x_real(i, refer(i) + 1) = xf(i)
                            v_real(i, refer(i) + 1) = vfin(i)
                            t_real(i, refer(i) + 1) = tf(i)

                            limit(i) = refer(i) + 2

                        End If

                    End If

                End If
```

```
            'Tiempo total del tramo y velocidad media
            '---------------------------------------

10          t_total(i) = ta(i) + tctt(i) + tf(i)
            v_mitjana(i) = lpol(i) / t_total(i)

        Next i

        t_recorrido = 0
        L_total = 0

        For j = 1 To n2
            t_recorrido = t_recorrido + t_total(j)
            L_total = L_total + lpol(j)
        Next j

        velo_recorrido = L_total / t_recorrido

        '-------------------------------------------------------------------------------------------------------------------
--

        'CONSTRUIMOS TRES VECTORES DE ESPACIO, VELOCIDAD Y TIEMPO CONJUNTOS DE TODO EL RECORRIDO
        'Aprovechamos las matrices x_real, v_real y t_real que hemos ido completando en cada tramo

        x_dib(0) = 0
        v_dib(0) = v0

        ac = 0 '(variable que recogerá los subíndices de los vectores ya utilizados)
        For i = 1 To n2
            For j = 1 To limit(i)
                x_dib(j + ac) = x_real(i, j - 1) + x_dib(j + ac - 1)
                v_dib(j + ac) = v_real(i, j - 1)
                t_dib(j + ac) = t_real(i, j - 1) + t_dib(j + ac - 1)
            Next j
            ac = ac + limit(i)
        Next i



        'PUBLICACIÓN DE RESULTADOS EN EL FORMULARIO

        'Velocidad de recorrido
'           ListVpolrec.AddItem Format(velo_recorrido * 3.6, "####0.0")
```

```
            'Tiempo de recorrido
            Dim minuts, segons As Integer
            minuts = Int(t_recorrido / 60)
            segons = t_recorrido - minuts * 60


'            ListTpolrec.AddItem Format(minuts, "####0")
'            ListSeg.AddItem Format(segons, "####0")

            'Recorrido de velocidades
            For i = 1 To ac
'               ListEspai.AddItem Format(x_dib(i), "####0.0") 'en m
'               ListVelo.AddItem Format(v_dib(i) * 3.6, "####0.0") 'en km/h
'               ListTemps.AddItem Format(t_dib(i), "####0.0") 'en segundos
            Next i


        End If

        objRecordset.Edit
        objRecordset.GFields("Vreco").Value = velo_recorrido * 3.6
        objRecordset.Update

        'PUBICACIÓN DE RESULTADOS EN UNA TABLA DE ACCES

'        If n > 2 Then
'            Call Publicació_Resultats
'        End If

        objRecordset.MoveNext

    Next elem
Else
     Call MsgBox("Cal seleccionar un objecte!", vbExclamation, App.ProductName)
     Hide
End If


'Explicitly release references
Set objSelectedObjects = Nothing
Set objDocument = Nothing

'Error handler
```

```
ErrorHandler:
    If Err.Number <> 0 Then
        Call MsgBox("Form_Load: " & Err.Description, vbOKOnly + vbExclamation, App.ProductName)
    End If

End Sub



Private Sub cmdEnd_Click()
Hide
End Sub


Private Sub Command1_Click()
Dim TaulaRadis As GRecordset

Set TaulaRadis = GMFeatureComboboxInput1.OutputRecordset
    TaulaRadis.MoveFirst
    TaulaRadis.Edit
    TaulaRadis.AddNew
    TaulaRadis.GFields("Radi").Value = 1
    TaulaRadis.Update
    TaulaRadis.Close


End Sub

Private Sub Form_Load()

Set GMFeatureComboboxInput1.Application = gobjGeoApp

End Sub




Private Sub Publicació_Resultats()

Dim TaulaRadis As GRecordset
Dim i As Long

Set TaulaRadis = GMFeatureComboboxInput1.OutputRecordset
```

```
frmMesures - 26

TaulaRadis.MoveFirst
TaulaRadis.Edit
TaulaRadis.AddNew


TaulaRadis.GFields("velo_recorrido").Value = velo_recorrido * 3.6
TaulaRadis.GFields("tiempo_recorrido").Value = t_recorrido



If elem = 100 Then
    MsgBox "he arribat al 100 i he escrit temps i velo recorr"
End If

If elem = 200 Then
    MsgBox "he arribat al 200 i he escrit temps i velo recorr"
End If

If elem = 300 Then
    MsgBox "he arribat al 300 i he escrit temps i velo recorr"
End If



If elem = 385 Then
    MsgBox "he arribat al 385 i he escrit temps i velo recorr"
End If

If elem = 386 Then
    MsgBox "he arribat al 386 i he escrit temps i velo recorr"
End If

If elem = 387 Then
    MsgBox "he arribat al 387 i he escrit temps i velo recorr"
End If

If elem = 388 Then
    MsgBox "he arribat al 388 i he escrit temps i velo recorr"
End If

If elem = 389 Then
    MsgBox "he arribat al 389 i he escrit temps i velo recorr"
End If
```

```
If elem = 390 Then
    MsgBox "he arribat al 390 i he escrit temps i velo recorr"
End If


TaulaRadis.Update
TaulaRadis.AddNew


For i = 1 To n - 2


    If elem = 388 Then
        If i = 1 Then
            MsgBox "he arribat al 388 i he entrat al bucle"
        End If
         If i = 2 Then
            MsgBox "elem 388 i començo i = 2"
        End If

        If i = n - 2 Then
            MsgBox "he acabat el bucle"
        End If

    End If

    TaulaRadis.GFields("Punt").Value = i
    TaulaRadis.GFields("X(m)").Value = X(i)
    TaulaRadis.GFields("Y(m)").Value = Y(i)

    'TaulaRadis.GFields("Radio_entero").Value = R(i)
    TaulaRadis.GFields("l_ini").Value = Lpoltot(i) - lpol(i)

    TaulaRadis.GFields("Lacum_poli").Value = Lpoltot(i)

    TaulaRadis.GFields("Tram").Value = i
    TaulaRadis.GFields("Radi(m)").Value = R(i)
    TaulaRadis.GFields("L_arc(m)").Value = L(i)
    TaulaRadis.GFields("L_pol(m)").Value = lpol(i)
    TaulaRadis.GFields("Ve(km/h)").Value = ve(i) * 3.6

   ' TaulaRadis.GFields("t_tram(s)").Value = t_total(i)
    'TaulaRadis.GFields("v_mitjana(km/h)").Value = v_mitjana(i) * 3.6
```

```
    TaulaRadis.Update
    TaulaRadis.AddNew
Next i

  If elem = 388 Then
            MsgBox "he acabat el bucle"
    End If

TaulaRadis.Close

End Sub

Function accel(D_ctt, v, i, A_ctt, Fv1, v1, FA_ctt, FA_vel, FF_c As Variant) As Double
accel = (A_ctt * (Fv1 - D_ctt * (v - v1)) - (FA_ctt + i + 12.96 * FA_vel * v ^ 2 / 4500)) / (100 * FF_c)

End Function
```