

# Índice

<b>1. Vectores de <math>\mathbb{R}^n</math> - V01</b>	<b>2</b>
1.1. Español . . . . .	2
1.1.1. Escena 1 - Definición de vector de $\mathbb{R}^n$ . . . . .	2
1.1.2. Escena 2 - Notación vectores de $\mathbb{R}^n$ . . . . .	3
1.1.3. Escena 3 - Selección de elementos de un vector de $\mathbb{R}^n$ . . . . .	5
1.2. Versión en inglés . . . . .	7
<b>2. Trozos comunes de código</b>	<b>7</b>
2.1. Carga de la librería Manim y NacAL . . . . .	7
2.2. Credits . . . . .	8
<b>3. Rodando: 1,2,3... ¡acción!</b>	<b>8</b>

# Lección 1 del curso - Vídeo 1

Marcos Bujosa

23 de abril de 2024

## 1. Vectores de $\mathbb{R}^n$ - V01

### 1.1. Español

#### 1.1.1. Escena 1 - Definición de vector de $\mathbb{R}^n$

```
1  <<Carga de la librería Manim y NacAL con AzureService>>
2
3  <<copyright>>
4
5  class L01_V01_E01_VectoresDefinicion(VoiceoverScene):
6      def construct(self):
7          self.set_speech_service( AzureService(voice="es-ES-AlvaroNeural" ) )
8
9          #self.set_speech_service(GTTSService(lang="es", tld="com"))
10         myTemplate = TextTemplate()
11         myTemplate.add_to_preamble(r"\usepackage{nacal}")
12
13         # Copyright lateral
14         <<copyrightLateral>>
15
16         # Portada
17         titulo = Title(r"Definición de vector de  $\mathbb{R}^n$  y notación",
18                       tex_template = myTemplate,
19                       font_size=70).set_color(BLUE)
20         self.play(Write(titulo))
21         self.wait(1.5)
22         self.play(FadeOut(titulo))
23
24         # Definición de vector
25         definicion = Tex("Un vector de ",
26                          r"$\mathbb{R}^n$",
27                          " es una \emph{lista ordenada} de $n$ números",
28                          tex_template = myTemplate,
29                          ).to_edge(UP).set_width(13)
30         with self.voiceover(text=r"Un vector de  $\mathbb{R}^n$  es una lista ordenada de números. ") as tracker:
31             self.add(definicion)
32
33         # Aclaración de la notación
34         with self.voiceover(text=r"La  $\mathbb{R}$  indica que los números son reales.") as tracker:
35             self.play(Circumscribe(definicion[1][0], fade_out=True), run_time=tracker.duration)
36
37         with self.voiceover(text=r"Y el superíndice  $n$  indica que la lista contiene  $n$  números.") as tracker:
38             self.play(Circumscribe(definicion[1][1], fade_out=True), run_time=tracker.duration)
39             self.wait(0.3)
40
41         with self.voiceover(text=r""que la lista sea ordenada
42                             significa que importa el orden en el que aparecen sus
43                             elementos.""") as tracker:
44             self.play(Circumscribe(definicion[2][5:18], fade_out=True), run_time=tracker.duration)
45
```

```

46     # Ejemplos
47     Ej = Tex(r"\textbf{Ejemplos:}",
48             tex_template = myTemplate,
49             font_size=50).set_color(GREEN).next_to(definicion, DOWN, aligned_edge=LEFT)
50     self.add(Ej)
51
52     d = nc.Vector([1,2,3], 'fila')
53     Ej1 = MathTex( d.latex(), "\ne", nc.Vector(reversed(d), 'fila').latex() )
54     with self.voiceover(text=r"Por ejemplo, los vectores, uno dos tres y tres dos uno, son distintos.") as tracker:
55         self.play(FadeIn(Ej1))
56         self.wait(tracker.duration-0.5)
57         self.play(FadeOut(Ej1))
58
59     p = nc.Vector([sp.pi, sp.pi, sp.pi, sp.pi], 'fila')
60     Ej2 = MathTex(p.latex(), r"\ne", nc.Vector(p[1,2], 'fila').latex() )
61     with self.voiceover(text=r""""También son distintos los vectores con distinta cantidad de
62     elementos. Por ejemplo, el vector de la izquierda pertenece a  $\mathbb{R}^4$  por ser una lista de
63     4 números. El de la derecha pertenece a  $\mathbb{R}^2$ .""") as tracker:
64         self.add(Ej2)
65         self.wait(tracker.duration-0.5)
66         self.play(FadeOut(Ej2))
67
68     c = nc.Vector([sp.pi, nc.frac(3,4), 0, 0.11], 'fila')
69     Ej3 = MathTex(c.latex(), "=", c.latex())
70     with self.voiceover(text=r""""En consecuencia, dos vectores serán iguales si, y solo si, sus correspondientes
71     listas son idénticas""") as tracker:
72         self.play(FadeIn(Ej3))
73         self.wait(tracker.duration)
74         self.play(FadeOut(Ej3))
75     self.play(FadeOut(Ej))
76     self.wait()
77

```

---

### 1.1.2. Escena 2 - Notación vectores de $\mathbb{R}^n$

```

class L01_V01_E02_VectoresNotacion(VoiceoverScene):
    def construct(self):
        self.set_speech_service( AzureService(voice="es-ES-AlvaroNeural" ) )

        #self.set_speech_service(GTTSService(lang="es", tld="com"))
        myTemplate = TextTemplate()
        myTemplate.add_to_preamble(r"\usepackage{nacal}")

        # Copyright lateral
        <<copyrightLateral>>

        # Definición de vector
        definicion = Tex("Un vector de ",
            r"$\mathbb{R}^n$",
            r" es una \emph{lista ordenada} de $n$ números",
            tex_template = myTemplate,
            ).to_edge(UP).set_width(13)
        self.add(definicion)
        d = nc.Vector([1,2,3], 'fila')
        p = nc.Vector([sp.pi, sp.pi, sp.pi, sp.pi], 'fila')

        # Notacion
        Notac = Tex(r"\textbf{Notación:}",
            tex_template = myTemplate,
            font_size=50).set_color(BLUE).next_to(definicion, DOWN, aligned_edge=LEFT)
        self.add(Notac)
        self.wait()

        a = nc.Vector( [5, 1, 10] , 'fila')
        Not1 = MathTex(a.latex(), tex_template = myTemplate,)
        Not2 = MathTex(r"=", tex_template = myTemplate,)

```

```

32 Not3 = MathTex(a.copy().latex(), tex_template = myTemplate,)
33 grp1 = VGroup(Not1,Not2,Not3).arrange(RIGHT)
34 with self.voiceover(text=r""""Para expresar un vector basta indicar la lista de elementos en su
35 correspondiente orden.""") as tracker:
36     self.add(grp1)
37     self.play(Circumscribe(definicion[2][5:18]),run_time=tracker.duration)
38
39 with self.voiceover(text=r""""Por este motivo podemos escribir un mismo vector tanto en
40 horizontal como en vertical. Pero tenga en cuenta que la
41 mayoría de manuales no respetan este convenio, y consideran,
42 al contrario de lo que haremos nosotros, que vectores fila y
43 vectores columna son objetos distintos.""") as tracker:
44     self.wait(tracker.duration+0.3)
45
46 with self.voiceover(text=r""""Siempre escribiremos la lista de números encerrada entre
47 paréntesis; poniendo una coma detrás de cada elemento cuando
48 escribamos el vector en horizontal.""") as tracker:
49     self.play(Indicate(grp1[0][0][:len(grp1[0][0])-1]),
50               Indicate(grp1[2][0][0:2]), Indicate(grp1[2][0][-2:]),
51               run_time=tracker.duration/2)
52     self.play(Flash(grp1[0][0][2]),
53               Flash(grp1[0][0][4]), Flash(grp1[0][0][7]),
54               run_time=tracker.duration/8)
55     self.wait(tracker.duration/8)
56     self.play(Circumscribe(grp1[0]))
57     self.play(FadeOut(grp1))
58
59 VectorNoNumero = MathTex(r"(3)",r"\ne", (3*nc.V1(1)).latex(),r"\in R[1]", tex_template = myTemplate,)
60 with self.voiceover(text=r""""Así podremos distinguir un número entre paréntesis de un
61 vector de  $R_1$ .""") as tracker:
62     self.add(VectorNoNumero)
63     self.play(Indicate(VectorNoNumero[0]),run_time=tracker.duration*2/3)
64     self.play(Indicate(VectorNoNumero[2]),run_time=tracker.duration/3)
65 with self.voiceover(text=r""""es decir, de una lista con un solo número.""") as tracker:
66     self.play(Indicate(VectorNoNumero[3]),
67               Flash(definicion[2][-9]),
68               run_time=tracker.duration)
69     self.play(FadeOut(VectorNoNumero))
70
71 Vectores = MathTex(r"\Vect{a}, \Vect{b}, \Vect{c}, \ldots \Vect{x}, \Vect{y}, \Vect{z}",
72                   tex_template = myTemplate,).move_to( UP )
73 Vector1 = MathTex(r"\Vect{a}=",a.copy().latex(), tex_template = myTemplate,)
74 Vector2 = MathTex(r"\Vect{d}=",d.copy().latex(), tex_template = myTemplate,)
75 Vector3 = MathTex(r"\Vect{x}=",p.copy().latex(), tex_template = myTemplate,)
76 grp3 = VGroup(Vector1,Vector2,Vector3).arrange(RIGHT, buff=2).next_to(Vectores, DOWN)
77 with self.voiceover(text=r""""Para denotar vectores usaremos letras minúsculas en negrita cursiva.""") as tracker:
78     self.add(Vectores)
79     self.add(grp3)
80     self.wait(tracker.duration/2)
81     self.play(Indicate(Vectores),run_time=tracker.duration/2)
82     self.play(FadeOut(Vectores))
83     self.play(Indicate(Vector1[0][0],scale_factor=2.),
84               Indicate(Vector2[0][0],scale_factor=2.),
85               Indicate(Vector3[0][0],scale_factor=2.),
86               run_time=1.5)
87     self.play(FadeOut(grp3))
88
89 Vnulo = MathTex(r"\Vect{0}", tex_template = myTemplate,).#move_to( UP )
90 with self.voiceover(text=r""""Un cero en negrita denota un vector cuyas componentes son todas nulas.""") as tracker:
91     self.add(Vnulo)
92     self.play(Indicate(Vnulo))
93     self.wait(tracker.duration/2)
94     self.play(FadeOut(Vnulo))
95
96 Vnulo1 = MathTex(r"\Vect{0}=", nc.V0(1).latex(), ",", tex_template = myTemplate,)
97 Vnulo2 = MathTex(r"\Vect{0}=", nc.V0(2).latex(), ",", tex_template = myTemplate,)
98 Vnulo3 = MathTex(r"\Vect{0}=", nc.V0(3).latex(), ",", tex_template = myTemplate,)
99 Vnulo6 = MathTex(r"\Vect{0}=", nc.V0(6).latex(), ",", tex_template = myTemplate,)

```

```

100     VnuloN = MathTex(r"\Vect{0}\in\mathbb{R}[100]", tex_template = myTemplate,)
101     grp2 = VGroup(Vnulo1, Vnulo2, Vnulo3, Vnulo6, VnuloN).arrange(RIGHT, buff=0.7)
102     with self.voiceover( text = r""Fijese que un cero en negrita
103     no indica su número de componentes. Normalmente la cantidad de
104     ceros se deduce del contexto."" ) as tracker:
105         self.add(grp2)
106         self.wait(tracker.duration)
107         self.play(FadeOut(grp2), FadeOut(Notac), FadeOut(definicion))
108         self.wait(1.5)

```

---

### 1.1.3. Escena 3 - Selección de elementos de un vector de $\mathbb{R}^n$

```

1  class L01_V01_E03_VectoresElementos(VoiceoverScene):
2      def construct(self):
3          self.set_speech_service( AzureService(voice="es-ES-AlvaroNeural" ) )
4
5          #self.set_speech_service(GTTSService(lang="es", tld="com"))
6
7          myTemplate = TextTemplate()
8          myTemplate.add_to_preamble(r"\usepackage{nacal}")
9
10         # Copyright lateral
11         <<copyrightLateral>>
12
13         # Notacion
14         Notac = Tex(r"\textbf{Notación para los elementos:}",
15                 tex_template = myTemplate,
16                 font_size=50).set_color(BLUE).to_corner(UL)
17         self.wait()
18         self.add(Notac)
19         self.wait()
20
21         # Elementos de un vector
22         v_generico = nc.Vector(sp.symbols('a:5')[1:], 'fila')
23         cs = MathTex(r"\Vect{a}=",
24                 v_generico.latex(),
25                 tex_template = myTemplate,)
26
27         with self.voiceover(text = r""Lo habitual es denotar cada
28         elemento de un vector con la letra de su nombre sin negrita."" ) as tracker:
29             self.wait()
30             self.play(FadeIn(cs), run_time=0.5)
31             self.play( Circumscribe(cs[1][1]),
32                     Circumscribe(cs[1][4]),
33                     Circumscribe(cs[1][7]),
34                     Circumscribe(cs[1][10]),
35                     run_time=tracker.duration/2)
36
37         with self.voiceover(text = r""indicando con un subíndice su posición en la lista."" ) as tracker:
38             self.play( Flash(cs[1][2]),
39                     Flash(cs[1][5]),
40                     Flash(cs[1][8]),
41                     Flash(cs[1][11]),
42                     run_time=tracker.duration)
43             self.play(FadeOut(cs))
44
45         c = nc.Vector([sp.pi, nc.frac(3,4), 0, 0.11], 'fila')
46         vector_c = MathTex(r"\Vect{c}=", c.latex(), tex_template = myTemplate,)
47         A = VGroup(*[ MathTex("c_"+str(i+1)+"="+sp.latex(e)) for i,e in enumerate(c.lista)
48                 ]).arrange(DOWN, aligned_edge=LEFT, buff=.5)
49         B = Brace(A, LEFT)
50         C = VGroup(A, B)
51         Elementos_c = VGroup(vector_c, C).arrange(RIGHT, buff=1)
52         with self.voiceover(text = r""Así, para el vector C """) as tracker:
53             self.play(FadeIn(vector_c))
54             self.play(GrowFromCenter(B), FadeIn(A))

```

```

55
56 with self.voiceover(text = r"""con c 1 denotamos su primera componente""") as tracker:
57     self.play( Indicate(vector_c[1][1]), Indicate(A[0]) )
58 with self.voiceover(text = r"""con c 2 la segunda""") as tracker:
59     self.play( Indicate(vector_c[1][3:6]), Indicate(A[1]) )
60 with self.voiceover(text = r"""y del mismo modo con el resto de componentes""") as tracker:
61     self.play( Indicate(vector_c[1][7], run_time=tracker.duration/2), Indicate(A[2], run_time=tracker.duration/2) )
62     self.play( Indicate(vector_c[1][9:13], run_time=tracker.duration/2), Indicate(A[3], run_time=tracker.duration/2) )
63     self.wait(0.5)
64     self.play( FadeOut(vector_c), FadeOut(B), FadeOut(A) )
65     self.wait(0.5)
66
67 with self.voiceover(text = r"""El hecho de emplear dos tipos
68 de fuentes: """) as tracker:
69     self.add(cs)
70     self.wait(tracker.duration)
71
72 with self.voiceover(text = r"""con negrita los vectores y sin negrita los
73 componentes, dificulta distinguirlos a primera vista""") as tracker:
74     self.play( Indicate(cs[0][ 0],scale_factor=2.),
75               Indicate(cs[0][ 0],scale_factor=2.),
76               Indicate(cs[1][ 1],scale_factor=2.),
77               Indicate(cs[1][ 4],scale_factor=2.),
78               Indicate(cs[1][ 7],scale_factor=2.),
79               Indicate(cs[1][10],scale_factor=2.), run_time=tracker.duration*2/3)
80
81 MTa = MathTex(r"\eleVR{a}{i}",tex_template = myTemplate).scale(3)
82 MTb = MathTex(r"{a}_{i}",tex_template = myTemplate).scale(3).next_to(MTa, LEFT)
83 VG = VGroup(MTb,MTa)
84 with self.voiceover(text = r"""Es más clara y operativa una notación que use un único tipo de fuente,
85 y que denote la selección de elementos con un operador. Por
86 ejemplo con una barra vertical.""") as tracker:
87     self.play(cs.animate.to_corner(DL),
88               run_time=tracker.duration*4/5)
89     self.play(Indicate(VG[1][0][1]))
90     self.wait(0.5)
91
92 def VectorGenerico(s,n):
93     elem = lambda s,i: sp.Symbol(r'\eleVR{' + s + '}' + str(i) + '}')
94     return nc.Vector([elem(s,i) for i in range(1,n+1)], 'fila')
95
96 v_generico2 = VectorGenerico('a',4)
97 cs2 = MathTex(r"=",
98               v_generico2.latex(),
99               tex_template = myTemplate,).next_to(cs, RIGHT)
100
101 VGB = VGroup(*[MathTex(sp.latex(e) + "=;" & \eleVR{a}{i} + str(i+1) + "}",
102                       tex_template = myTemplate)
103               for i,e in enumerate(v_generico2.lista)
104               ]).scale(3)
105
106 with self.voiceover( text = r"""Por ello, para denotar una componente, escribiremos un subíndice con una
107 barra que medie entre el vector y el índice de la
108 componente""") as tracker:
109     self.play(FadeIn(VG[1]))
110     self.wait(tracker.duration/3)
111     self.play(Indicate(VG[1][0][1:], run_time=tracker.duration/4))
112     #self.wait(tracker.duration/3)
113     self.play(Indicate(VG[1][0][-1], run_time=tracker.duration/5))
114     self.play(Write(VG[0]))
115     self.wait()
116     self.play(VG.animate.move_to([0,0,0]))
117     self.play(Transform(VG[1][0][-1],VGB[0][0][-1]),
118               Transform(VG[0][0][:2],VGB[0][0][:2]), run_time=1.5)
119     self.play( FadeIn(cs2) )
120     self.play(FadeTransform(VGB[0][0][0:2],cs[1][ 1: 3]),
121               FadeTransform(VGB[0][0][3:],cs2[1][ 1: 6]), FadeOut(VG), run_time=1.5)
122     self.play(FadeIn(VGB[1]), FadeOut(VGB[0][0][2]))

```

```

123         self.play(FadeTransform(VGB[1][0][0:2],cs[1][4:6]),
124                   FadeTransform(VGB[1][0][3:],cs2[1][7:12]), FadeOut(VGB[1][0][2]), run_time=1.5)
125         self.play(FadeIn(VGB[2]))
126         self.play(FadeTransform(VGB[2][0][0:2],cs[1][7:9]),
127                   FadeTransform(VGB[2][0][3:],cs2[1][13:18]), FadeOut(VGB[2][0][2]), run_time=1.5)
128         self.play(FadeIn(VGB[3]))
129         self.play(FadeTransform(VGB[3][0][0:2],cs[1][10:12]),
130                   FadeTransform(VGB[3][0][3:],cs2[1][19:24]), FadeOut(VGB[3][0][2]), run_time=1.5)
131         self.play(FadeOut(Notac),FadeOut(cs),FadeOut(cs2))
132
133         MTLR = MathTex(r"\eleVR{a}{i}",r"\;=\eleVL{a}{i}",tex_template = myTemplate).scale(3)
134         with self.voiceover( text = r""""Además, admitiremos que el operador selector actúe tanto por la derecha
135         como por la izquierda.""") as tracker:
136             self.play(FadeIn(MTLR[0]), run_time=2*tracker.duration/3)
137             self.play(FadeIn(MTLR[1]))
138             self.wait(tracker.duration/3+0.5)
139             self.play(FadeOut(MTLR))
140             self.wait()

```

---

## 1.2. Versión en inglés

```

1 manim_render_translation $fichero.py -s $escena -d $escenaENG -l en -ql

```

---

## 2. Trozos comunes de código

### 2.1. Carga de la librería Manim y NacAL

---

```

1 from manim import *
2 from manim_voiceover import VoiceoverScene
3 from manim_voiceover.services.gtts import GTTSService
4 import nacal as nc
5 import sympy as sp
6
7 # PARA LA TRADUCCIÓN (pero no me ha funcionado)
8
9 #from manim_voiceover.translate import get_gettext
10 # # It is good practice to get the LOCALE and DOMAIN from environment variables
11 #import os
12 #LOCALE = os.getenv("LOCALE")
13 #DOMAIN = os.getenv("DOMAIN")
14 # The following function uses LOCALE and DOMAIN to set the language, and
15 # returns a gettext function that is used to insert translations.
16 #_ = get_gettext()

```

---



---

```

1 from manim import *
2 from manim_voiceover import VoiceoverScene
3 from manim_voiceover.services.azure import AzureService
4 import nacal as nc
5 import sympy as sp
6
7 # PARA LA TRADUCCIÓN (pero no me ha funcionado)
8
9 #from manim_voiceover.translate import get_gettext
10 # # It is good practice to get the LOCALE and DOMAIN from environment variables
11 #import os
12 #LOCALE = os.getenv("LOCALE")
13 #DOMAIN = os.getenv("DOMAIN")
14 # The following function uses LOCALE and DOMAIN to set the language, and
15 # returns a gettext function that is used to insert translations.
16 #_ = get_gettext()

```

---

## 2.2. Créditos

```
1 copyright = Tex(r"\textcopyright{\;} 2024\; Marcos Bujosa ")
2 CGG = VGroup(copyright).rotate(PI/2).scale(0.5).to_edge(RIGHT).set_color(GRAY_D)
3 self.add(CGG)

class ZCreditos(Scene):
    def construct(self):
        copyright = Tex(r"\textcopyright{\;} 2024 \; Marcos Bujosa")
        github = Tex(r"\texttt{https://github.com/mbujosab}").next_to(copyright, DOWN)
        CGG = VGroup(copyright, github).scale(1.1)
        self.add(CGG)
        self.wait(10)
```

## 3. Rodando: 1,2,3... ¡acción!

1. Generamos un fichero mpeg por cada escena

```
1 echo $escena | manim -pql $fichero.py --disable_caching

1 echo $escena | manim -qh $fichero.py --disable_caching
```

2. Concatenamos las escenas en un único fichero mpeg y añadimos música de fondo.

- Montando la versión de baja resolución

```
1 rm -f $subdir/$video/$calidad/$video.mp4 list.txt
2 for f in $subdir/$video/$calidad/*.mp4 ; do echo file '$f' >> list.txt; done && ffmpeg -f concat -safe 0 -i list.txt -c
3
4 mkdir -p tmp
5
6 ffmpeg -i $subdir/$video/$calidad/$video.mp4 -i $music.mp3 -filter_complex "[0:a]apad[main]; [1:a]volume=0.04,apad[A]; [m
```

- a) La primera parte de este script concatena todos los archivos de video .mp4 encontrados en una carpeta específica en un solo archivo de video. Lo hace en dos pasos:

- **Genera un archivo de lista de archivos (list.txt):**

- Utiliza un bucle `for` para iterar sobre todos los archivos con extensión .mp4 en la carpeta especificada por la variable `$subdir/$video/$calidad/`.
- Por cada archivo encontrado, agrega una línea al archivo `list.txt` que contiene la ruta completa del archivo.
- Utiliza el operador `>>` para redirigir la salida al final del archivo `list.txt`.

- **Utiliza ffmpeg para concatenar los archivos de video listados en list.txt:**

- Utiliza `ffmpeg` con la opción `-f concat` para indicar que se está utilizando un archivo de lista para concatenar.
- `-safe 0` se usa para permitir la lectura de archivos de cualquier ruta.
- `-i list.txt` especifica el archivo de lista generado anteriormente como entrada.
- `-c copy` indica a `ffmpeg` que simplemente copie los flujos de audio y video de entrada sin realizar ninguna conversión.



- Finalmente, el resultado se guarda en un nuevo archivo de video con el mismo nombre que la variable `$video`, pero con la extensión `.mp4`, en la carpeta especificada por `$subdir/$video/$calidad/`.
- b) En la segunda parte el comando de ‘ffmpeg’ realiza varias acciones para procesar un video junto con un archivo de audio y generar un nuevo archivo de video:

1) **Entradas de archivos de video y audio:**

- `-i $subdir/$video/$calidad/$video.mp4`: Especifica el archivo de video de entrada.
- `-i $music.mp3`: Especifica el archivo de audio de entrada que se usará como música de fondo.

2) **Filtro de complejidad (`-filter_complex`):**

- `[0:a]apad[main]`: Añade un silencio al audio del video de entrada (`[0:a]`) y lo asigna a la etiqueta `[main]`.
- `[1:a]volume=0.04,apad[A]`: Ajusta el volumen del audio de entrada (`[1:a]`) al 4 % del volumen original, añade un silencio y lo asigna a la etiqueta `[A]`.
- `[main][A]amerge[out]`: Fusiona los flujos de audio etiquetados como `[main]` y `[A]` en un solo flujo de audio, asignado a la etiqueta `[out]`.

3) **Configuración de códecs y opciones de salida:**

- `-c:v libx264`: Utiliza el códec de video H.264 para codificar el video de salida.
- `-c:a aac`: Utiliza el códec de audio AAC para codificar el audio de salida.
- `-map 0:v -map "[out]"`: Mapea el flujo de video del archivo de video de entrada y el flujo de audio resultante del filtro complejo.
- `-preset ultrafast`: Establece el preset de codificación en ‘ultrafast’ para una codificación más rápida.
- `-threads 0`: Utiliza el número máximo de hilos disponibles para la codificación.
- `-profile:v baseline`: Especifica el perfil de codificación de video como ‘baseline’.
- `-ac 2`: Establece el número de canales de audio a 2 (estéreo).
- `-pix_fmt yuv420p`: Especifica el formato de píxeles del video de salida como yuv420p.
- `-shortest`: Hace que la duración del video de salida sea igual a la duración del archivo de video de entrada más corto o la duración del audio de entrada más corto.
- `-y tmp/$video.mp4`: Especifica el nombre del archivo de video de salida y la ruta donde se guardará.

En resumen, este comando de `ffmpeg` combina un video con un archivo de audio de fondo, ajustando el volumen del audio de fondo y fusionando los flujos de audio para crear un nuevo archivo de video con las especificaciones dadas.

- Montando la versión de resolución HD1080

---

```

1  rm -f $subdir/$video/$calidad/$video.mp4 list.txt
2  for f in $subdir/$video/$calidad/*.mp4 ; do echo file '$f' >> list.txt; done && ffmpeg -f concat -safe 0 -i list.txt -c
3
4  mkdir -p tmp
5
6  ffmpeg -i $subdir/$video/$calidad/$video.mp4 -i $music.mp3 -filter_complex "[0:a]apad[main]; [1:a]volume=0.04,apad[A]; [m

```

---

3. Fundimos a negro los últimos segundos del vídeo (y la música).

---

```
1 dur=$(ffprobe -loglevel error -show_entries format=duration -of default=nk=1:nw=1 "tmp/$video.mp4") && offset=$(bc -l <<<
```

---

Este código es un script que utiliza las herramientas `ffprobe` y `ffmpeg` para aplicar efectos de fundido a un video. Aquí está desglosado paso a paso:

- a) `dur=$(ffprobe -loglevel error -show_entries format=duration -of default=nk=1:nw=1 "tmp/$video.mp4")`: Este comando utiliza `ffprobe` para obtener la duración del video especificado por la variable `$video`. La duración se guarda en la variable `dur`.
- b) `offset=$(bc -l \<\<\<"$dur7)`: Luego, se calcula un valor de desplazamiento (`offset`) restando 7 segundos de la duración del video. Esto sugiere que el objetivo es aplicar los efectos de fundido durante los últimos 7 segundos del video.
- c) `ffmpeg -i "tmp/$video.mp4" filter_complex "[0:v]fade=type=in:duration=0,fade=type=out:duration=$offset[v];[0:a]afade=type=in:duration=1,afade=type=out:duration=6:start_time=$offset[a]map "[v]map "[a]` y `$video.mp4`: Finalmente, se utiliza `ffmpeg` para aplicar los efectos de fundido. El filtro `-filter_complex` se utiliza para aplicar un fundido de entrada (`fade=type=in`) y un fundido de salida (`fade=type=out`) tanto al video como al audio. El fundido de entrada del audio (`afade=type=in:duration=1`) tiene una duración de 1 segundo. Los fundidos de salida de audio y video tienen una duración de 6 segundos y comienzan en el tiempo especificado por el `offset`. El resultado se guarda en un nuevo archivo de video con el nombre `$video.mp4`.

En resumen, este script toma un video, calcula su duración, y luego aplica efectos de fundido de entrada y salida tanto al video como al audio, comenzando 7 segundos antes del final del video.

4. Copiamos el resultado a un lugar público

---

```
1 cp -f $video.mp4 $subdir/$video.mp4
```

---