

Índice

0.1. Definición una subclase con algunas definiciones generales	2
1. Suma de Vectores de \mathbb{R}^n - V02	3
1.1. Español	3
1.1.1. Escena 1 - Suma de vectores de \mathbb{R}^n	3
1.1.2. Escena 2 - Propiedad conmutativa de la suma	6
1.1.3. Escena 3 - Interpretación geométrica de la suma en \mathbb{R}^2	8
1.1.4. Escena 4 - Interpretación geométrica de la suma en \mathbb{R}^3	15
1.1.5. Escena 5 - Interpretación geométrica de la suma en \mathbb{R}^n	18
1.1.6. Escena 6 - Resumen	20
2. Trozos comunes de código	21
2.1. Carga de la librería Manim y NacAL	21
2.2. Credits	22
3. Rodando: 1,2,3... ¡acción!	22

Lección 1 del curso - Vídeo 2

Marcos Bujosa

8 de mayo de 2024

0.1. Definición una subclase con algunas definiciones generales

```
1 class MiEscenaConVoz(VoiceoverScene):
2     def pausa_muy_corta(self, n=0.3):
3         self.wait(n)
4
5     def pausa_corta(self, n=0.5):
6         self.wait(n)
7
8     def pausa(self, n=1):
9         self.wait(n)
10
11     def pausa_media(self, n=1.5):
12         self.wait(n)
13
14     def pausa_larga(self, n=3):
15         self.wait(n)
16
17     def pausa_muy_larga(self, n=5):
18         self.wait(n)
19
20     <<Créditos en distintas partes de la pantalla>>
```

```
1 def creditos(self, variante=1):
2     def analisis_opcion_elegida(tipo):
3         'Análisis de las opciones de eliminación elegidas'
4         lista = [100,20,10,4,2,1]
5         opcion = set()
6         for t in lista:
7             if (tipo - (tipo % t)) in lista:
8                 opcion.add(tipo - (tipo % t))
9                 tipo = tipo % t
10        return opcion
11
12    copyright = Tex(r"\textcopyright{\;} 2024\; Marcos Bujosa ")
13    if 1 in analisis_opcion_elegida(variante):
14        stampDcha = VGroup(copyright.copy()).rotate( PI/2).scale(0.5).to_edge(RIGHT, buff=0.1).set_color(GRAY_D)
15        self.add(stampDcha)
16    if 2 in analisis_opcion_elegida(variante):
17        stampIzda = VGroup(copyright.copy()).rotate(-PI/2).scale(0.5).to_edge(LEFT, buff=0.1).set_color(GRAY_D)
18        self.add(stampIzda)
19    if 4 in analisis_opcion_elegida(variante):
20        stampBottom= VGroup(copyright.copy()).rotate( 0).scale(0.5).to_edge(DOWN, buff=0.1).set_color(GRAY_D)
21        self.add(stampBottom)
22    if 10 in analisis_opcion_elegida(variante):
23        stampTop = VGroup(copyright.copy()).rotate( 0).scale(0.5).to_edge( UP, buff=0.1).set_color(GRAY_D)
24        self.add(stampTop)
```

```
1 class VectorR2():
2     def __init__(self, lista, rpr='c', color=GRAY_B):
```

```

3      """Inicializa un Vector con una lista"""
4      coords = lista + [0] if len(lista)<3 else lista
5      self.color = color
6      self.coords = tuple(coords)
7      self.Vector = nc.Vector(self.coords[:2], rpr)
8      self.tex = MathTex(self.Vector.latex(), color=self.color).scale(0.8)
9
10     def dot(self, ejes, radio=0.08):
11         return Dot(ejes.c2p(*self.coords), radius=radio, color=self.color)
12
13     def v_line(self, ejes):
14         return ejes.get_vertical_line(ejes.c2p(*self.coords), color=self.color)
15
16     def h_line(self, ejes):
17         return ejes.get_horizontal_line(ejes.c2p(*self.coords), color=self.color)
18
19     def arrow(self, ejes):
20         return ejes.get_vector(self.Vector.lista, stroke_color = self.color, stroke_width=4)

```

```

1 class VectorR3():
2     def __init__(self, lista, rpr='c', color=GRAY_B):
3         """Inicializa un Vector con una lista"""
4         coords = lista + [0] if len(lista)<3 else lista
5         self.color = color
6         self.coords = tuple(coords)
7         self.Vector = nc.Vector(self.coords, rpr)
8         self.tex = MathTex(self.Vector.latex(), color=self.color).scale(0.8)
9
10    def dot(self, ejes, radio=0.08):
11        return Dot3D(ejes.c2p(*self.coords), radius=radio, color=self.color)
12
13    def v_line(self, ejes):
14        return ejes.get_vertical_line(ejes.c2p(*self.coords), color=self.color)
15
16    def h_line(self, ejes):
17        return ejes.get_horizontal_line(ejes.c2p(*self.coords), color=self.color)
18
19    def arrow(self, ejes):
20        return Arrow3D(
21            start=np.array([0, 0, 0]),
22            end=np.array(ejes.c2p(*self.coords)),
23            resolution=8,
24            color = self.color )

```

1. Suma de Vectores de \mathbb{R}^n - V02

1.1. Español

1.1.1. Escena 1 - Suma de vectores de \mathbb{R}^n

```

1 <<Carga de la librería Manim y NacAL con AzureService>>
2 <<Definición de mi escena con voz>>
3 <<Definición de VectorR2>>
4 <<Definición de VectorR3>>
5
6 import itertools
7 def get_sub_indexes(tex):
8     ni = VGroup()
9     colors = itertools.cycle([RED,TEAL,GREEN,BLUE,PURPLE])
10    for i in range(len(tex)):
11        n = Text(f"{i}",color=next(colors)).scale(0.7)
12        n.next_to(tex[i],DOWN,buff=0.01)
13        ni.add(n)

```

```

14     return ni
15
16 class L01_V02_E01_SumaDeVectores(MiEscenaConVoz):
17     def construct(self):
18         self.set_speech_service( AzureService(voice="es-ES-AlvaroNeural" ) )
19         #self.set_speech_service(GTTSService(lang="es", tld="com"))
20
21         myTemplate = TextTemplate()
22         myTemplate.add_to_preamble(r"""\usepackage{nacal} """)
23
24         self.creditos()
25
26         # Portada
27         titulo = Title(r"Suma de vectores de  $\mathbb{R}^n$ ",
28                       tex_template = myTemplate,
29                       font_size=70).set_color(BLUE)
30         self.play(Write(titulo))
31         self.pausa_media()
32         self.play(FadeOut(titulo))
33
34         # Definición de vector suma
35         operacionSuma = Tex(r"Suma de vectores de  $\mathbb{R}^n$ ",
36                           tex_template = myTemplate, font_size=70
37                           ).to_edge(UP).set_color(BLUE)
38
39         operacionDescripcion = Tex("La suma se define componente a componente.",
40                                   tex_template = myTemplate,
41                                   ).move_to([0,2.5,0]).to_edge(LEFT)
42
43         # Ejemplos
44         EjR3 = Tex(r"\textbf{Ejemplo en  $\mathbb{R}^3$ :}",
45                  tex_template = myTemplate,
46                  font_size=50).set_color(GREEN).next_to(operacionDescripcion, DOWN, aligned_edge=LEFT)
47
48         EjR4 = Tex(r"\textbf{Ejemplo en  $\mathbb{R}^4$ :}",
49                  tex_template = myTemplate,
50                  font_size=50).set_color(GREEN).next_to(operacionDescripcion, DOWN, aligned_edge=LEFT)
51
52         a = nc.Vector( [0, 3, 6])
53         b = nc.Vector( [5, 1, 2])
54         s1 = MathTex(a.latex(), tex_template = myTemplate,)
55         mas= MathTex(r"+", tex_template = myTemplate,)
56         s3 = MathTex(b.latex(), tex_template = myTemplate,)
57         igual = MathTex(r "=", tex_template = myTemplate,)
58         s5 = MathTex((a+b).latex(), tex_template = myTemplate,)
59         grp1 = VGroup(s1,mas,s3,igual,s5).arrange(RIGHT)
60
61         self.add(operacionSuma)
62         self.add(operacionDescripcion)
63
64         with self.voiceover(text=r"""\Podemos sumar dos vectores si ambos poseen el mismo número de
65         componentes.""") as tracker:
66             self.add(EjR3)
67             self.add(grp1[0])
68             self.add(grp1[2])
69
70         # Definición de vector suma
71         with self.voiceover(text=r"""\El resultado es otro vector que se define componente a
72         componente.""") as tracker:
73             self.add(grp1[1])
74             self.pausa_media()
75             self.add(grp1[3])
76             self.add(grp1[4][0][:2])
77             self.add(grp1[4][0][-2:])
78
79         with self.voiceover(text=r"""\La primera es la suma de las primeras componentes de ambos
80         vectores.""") as tracker:
81             self.play(FadeIn(grp1[4][0][2]), run_time=tracker.duration/3)
82             self.play(Circumscribe(grp1[0][0][2]), Circumscribe(grp1[2][0][2]), run_time=tracker.duration*2/3)

```

```

82         self.pausa_corta()
83
84     with self.voiceover(text=r""""La segunda es la suma de las segundas.""") as tracker:
85         self.play(FadeIn(grp1[4][0][3]), run_time=tracker.duration/3)
86         self.play(Circumscribe(grp1[0][0][3]), Circumscribe(grp1[2][0][3]), run_time=tracker.duration*2/3)
87         self.pausa_corta()
88
89     with self.voiceover(text=r""""Y así con todas las componentes
90 del vector suma.""") as tracker:
91         self.play(FadeIn(grp1[4][0][4]))
92         self.pausa_corta()
93         self.play(Circumscribe(grp1[0][0][4]), Circumscribe(grp1[2][0][4]) )
94         self.pausa_corta(.3)
95         self.play(FadeOut(grp1))
96
97     self.pausa()
98
99     v_generico_a = nc.Vector(sp.symbols('a:5')[1:])
100    vga = MathTex(v_generico_a.latex(), tex_template = myTemplate)
101
102    v_generico_b = nc.Vector(sp.symbols('b:5')[1:])
103    vgb = MathTex(v_generico_b.latex(), tex_template = myTemplate)
104
105    vgab = MathTex((v_generico_a + v_generico_b).latex(), tex_template = myTemplate)
106
107    grp2 = VGroup(vga,mas,vgb,igual,vgab).arrange(RIGHT)
108    with self.voiceover(text=r""""Por tanto, la siguiente expresión describe la suma de vectores en R
109 4. """) as tracker:
110        self.play(FadeTransform(EjR3,EjR4))
111        self.play(FadeIn(grp2))
112        self.pausa()
113
114    with self.voiceover(text=r""""Definir la suma en R n requiere una estrategia distinta; una que no
115 necesite escribir la lista completa de componentes. Piense que
116 la lista puede ser muy larga para enes grandes.""") as tracker:
117        self.wait(tracker.duration/3)
118        self.play(Indicate(vga[0][4:-4]), Indicate(vgb[0][4:-4]), Indicate(vgab[0][4:-4]), run_time=2)
119        self.pausa_corta()
120
121    Defn = Tex(r"\textbf{Definición:}",
122              tex_template = myTemplate,
123              font_size=50).set_color(RED).next_to(operacionDescripcion, DOWN, aligned_edge=LEFT)
124
125    with self.voiceover(text=r""""Una solución es definir la suma usando la notación
126 descrita en el video anterior. Con ella podemos expresar""") as tracker:
127        self.play(FadeOut(grp2), FadeOut(EjR4), run_time=tracker.duration/3)
128        self.add(Defn)
129
130    cvab = MathTex(r"\elemRp{\Vect{a}+\Vect{b}}{i}", tex_template = myTemplate)
131    cva = MathTex(r"\eleVR{a}{i}", tex_template = myTemplate)
132    cvb = MathTex(r"\eleVR{b}{i}", tex_template = myTemplate)
133    eq_suma = VGroup(cvab,igual,cva,mas,cvab).arrange(RIGHT).scale(1.5)
134
135    donde = Tex("donde")
136    indices = MathTex(r"i=1:n", tex_template = myTemplate)
137    pc_indices = VGroup(donde,indices).arrange(RIGHT, buff=1)
138    grp3 = VGroup(eq_suma, pc_indices).arrange(RIGHT, buff=1)
139
140    with self.voiceover(text=r""""que la componente i-ésima del vector suma es igual a la suma de las i-ésimas componentes de
141 self.play(FadeIn(grp3[0][2:], scale=1.5, rate_func=rate_functions.exponential_decay), run_time=2*tracker.duration/5)
142 self.play(FadeIn(grp3[0][2:], scale=0.5, rate_func=rate_functions.exponential_decay), run_time=3*tracker.duration/5)
143 with self.voiceover(text=r""""(donde el índice recorre los números naturales entre uno y n)""") as tracker:
144     self.play(FadeIn(grp3[1]))
145     self.pausa_corta()
146
147    with self.voiceover(text=r""""Esta definición abstracta será muy util para demostrar algunas
148 propiedades de las operaciones con vectores, pues arroja una
149 primera regla de cálculo simbólico:""") as tracker:

```

```

150         self.pausa(tracker.duration*2/3)
151         self.play(Indicate(eq_suma[0][0][0]), Indicate(eq_suma[0][0][-3:]), Indicate(eq_suma[2][0][-2:]), Indicate(eq_suma[4][0][-2:]))
152
153
154         with self.voiceover(text=r""que la suma de las i ésimas componentes se puede sustituir por la
155         i-ésima componente del vector suma.""") as tracker:
156             source0 = MathTex(r"\eleVR{a}{i}+\eleVR{b}{i}", tex_template = myTemplate).next_to(grp3, DOWN, buff=1.2).scale(2)[0]
157             target0 = MathTex(r"\elemRp{\Vect{a}+\Vect{b}}{i}", tex_template = myTemplate).next_to(grp3, DOWN, buff=1.2).scale(2)[0]
158             source1 = target0.copy()
159             target1 = source0.copy()
160
161             VGroup(source0,target0)
162             self.add(source0)
163             transform_index0 = [
164                 [0,1,2,3,4,5,6],
165                 [1,0,4,2,3,5,6]
166             ]
167             self.play(
168                 *[
169                     ReplacementTransform(source0[i],target0[j], rate_func=rate_functions.smooth)
170                     for i,j in zip(*transform_index0)
171                 ],
172                 run_time=tracker.duration)
173
174         with self.voiceover(text=r""Y la i-ésima componente de una suma se puede sustituir por la
175         suma de las i ésimas componentes.""") as tracker:
176             self.play(ReplacementTransform(target0,source1))
177
178             VGroup(source1,target1)
179             transform_index1 = [
180                 [0,1,2,3,4,5,6],
181                 [1,0,3,4,2,5,6]
182             ]
183             self.play(
184                 *[
185                     ReplacementTransform(source1[i],target1[j], rate_func=rate_functions.smooth)
186                     for i,j in zip(*transform_index1)
187                 ],
188                 run_time=tracker.duration)
189
190         with self.voiceover(text=r""Esta regla se denomina propiedad distributiva del operador selector
191         respecto de la suma.""") as tracker:
192             self.play(FadeOut(target1))
193             self.play(Indicate(eq_suma[0][0][0]), Indicate(eq_suma[0][0][-3:]), Indicate(eq_suma[2][0][-2:]), Indicate(eq_suma[4][0][-2:]))
194             self.pausa()

```

1.1.2. Escena 2 - Propiedad conmutativa de la suma

```

1 class L01_V02_E02_PropiedadConmutativaDeLaSuma(MiEscenaConVoz):
2     def construct(self):
3         self.set_speech_service( AzureService(voice="es-ES-AlvaroNeural" ) )
4         #self.set_speech_service(GTTSService(lang="es", tld="com"))
5
6         myTemplate = TexTemplate()
7         myTemplate.add_to_preamble(r""\usepackage{nacal} "")
8
9         self.creditos(7)
10
11         # Definición de vector suma
12         operacionSuma = Tex(r"Suma de vectores de  $\mathbb{R}[n]$ ",
13                             tex_template = myTemplate, font_size=70
14                             ).to_edge(UP).set_color(BLUE)
15
16         self.add(operacionSuma)
17
18         str0 = MathTex(r"\elemRp{\Vect{a}+\Vect{b}}{i}", tex_template = myTemplate).scale(2)[0]

```

```

19 str1 = MathTex(r"\elemRp{\Vect{b}+\Vect{a}}{i}", tex_template = myTemplate).scale(2)[0]
20 igual = MathTex(r"=", tex_template = myTemplate).scale(2)[0]
21 vgr1 = VGroup(str0, igual, str1).arrange(RIGHT, buff=1)
22
23 with self.voiceover(text=r""Antes de continuar, demosremos la propiedad conmutativa de la suma
24 de vectores. Es decir, que el orden en que se sumen los
25 vectores es irrelevante.""") as tracker:
26     self.play(FadeIn(str0[1:-3]), FadeIn(vgr1[1]), FadeIn(str1[1:-3]))
27     self.pausa(tracker.duration/2)
28     self.play(Indicate(str0[1]), Indicate(str1[-4]), run_time=tracker.duration/4)
29     self.play(Indicate(str0[-4]), Indicate(str1[1]), run_time=tracker.duration/4)
30     self.pausa_corta()
31
32 with self.voiceover(text=r""Sabemos que dos vectores son iguales si lo son sus correspondientes
33 listas de componentes. Por tanto, para demostrar la igualdad
34 entre vectores debemos probar la igualdad componente a
35 componente.""") as tracker:
36     self.pausa(tracker.duration*2/5)
37     self.play(FadeIn(str0[0]), FadeIn(str0[-3:]), FadeIn(str1[0]), FadeIn(str1[-3:]))
38     self.pausa(tracker.duration/4)
39     self.play(Indicate(str0[-2:]), Indicate(str1[-2:]), run_time=tracker.duration/4)
40     self.pausa_corta()
41
42 with self.voiceover(text=r""Para ello comenzaremos escribiendo uno cualquiera de
43 sus lados. Después operaremos hasta obtener la expresión del
44 lado opuesto de la igualdad.""") as tracker:
45     self.pausa(tracker.duration/4)
46     self.play(Indicate(vgr1[0]))
47     self.play(Indicate(vgr1[2]))
48
49 vgr2=vgr1.copy().scale(1/2).next_to(operacionSuma, DOWN).to_edge(LEFT)
50 vgr3=vgr1.copy().scale(1/2).to_edge(LEFT)
51 item1 = MathTex(r"\eleVR{x}{i} \in \R", tex_template = myTemplate)
52 item2 = MathTex(r"\alpha + \beta = \beta + \alpha \quad (\alpha, \beta \in \R)", tex_template = myTemplate)
53 item3 = MathTex(r"\elemRp{\Vect{x}+\Vect{y}}{i} = \eleVR{x}{i} + \eleVR{y}{i}", tex_template = myTemplate)
54 items = VGroup(item1, item2, item3).arrange(DOWN).scale(.8).align_to(vgr2, UP).to_edge(RIGHT).shift(DOWN*0.15)
55 box = SurroundingRectangle(items, color=YELLOW)
56
57 paso1 = MathTex(r"\eleVR{a}{i}+\eleVR{b}{i}", tex_template = myTemplate).next_to(vgr3[0], RIGHT)
58 paso2 = MathTex(r"\eleVR{b}{i}+\eleVR{a}{i}", tex_template = myTemplate).next_to(paso1, DOWN, aligned_edge=LEFT)
59 paso3 = MathTex(r"\elemRp{\Vect{b}+\Vect{a}}{i}", tex_template = myTemplate).next_to(paso2, DOWN, aligned_edge=LEFT)
60 demo = VGroup(paso1, paso2, paso3)
61
62 with self.voiceover(text=r""Con operar nos referimos a sustituir una expresión por otra que
63 sabemos que es equivalente. Para esta demostración solo necesitamos considerar tres cosas""") as tracker:
64     self.play(FadeTransformPieces(vgr1, vgr2), run_time=tracker.duration/2)
65     self.add(box, items)
66     self.pausa_muy_larga()
67
68 with self.voiceover(text=r""que los elementos de un vector son números reales, que entre
69 números reales la suma es conmutativa, y que el operador
70 selector es distributivo respecto de la suma""") as tracker:
71     self.play(Indicate(items[0]), run_time=tracker.duration/3)
72     self.play(Indicate(items[1]), run_time=tracker.duration/3)
73     self.play(Indicate(items[2]), run_time=tracker.duration/3)
74
75 with self.voiceover(text=r""Comencemos escribiendo uno de los lados, por ejemplo el izquierdo.""") as tracker:
76     self.play(FadeTransformPieces(vgr2[0].copy(), vgr3[0]), FadeToColor(vgr2[0], color=TEAL), run_time=tracker.duration)
77     self.pausa_medio()
78
79 with self.voiceover(text=r""En primer lugar, el operador selector es distributivo respecto de la suma""") as tracker:
80     self.play(Indicate(items[2]), run_time=tracker.duration/2)
81     self.play(FadeIn(demo[0]), run_time=tracker.duration/2)
82
83 with self.voiceover(text=r""En segundo lugar, dado que los componentes son números reales, el
84 resultado no cambia si intercambiamos el orden de su suma.""") as tracker:
85     self.play(Indicate(items[:2]), run_time=tracker.duration/2)
86     self.play(FadeIn(demo[1]), run_time=tracker.duration/2)

```

```

87
88     with self.voiceover(text=r""""Por último, el operador selector es distributivo respecto de la suma""") as tracker:
89         self.play(Indicate(items[2], run_time=tracker.duration/2) )
90         self.play(FadeIn(demo[2], run_time=tracker.duration/2) )
91
92     with self.voiceover(text=r""""Con esto hemos terminado la demostración.""") as tracker:
93         self.play(FadeToColor(vgr2[0], color=TEAL))
94         self.play(Indicate(vgr3[0]), Indicate(demo[2]), FadeToColor(vgr2[1:], color=TEAL), run_time=tracker.duration)
95         self.pausa()

```

1.1.3. Escena 3 - Interpretación geométrica de la suma en \mathbb{R}^2

```

1  class L01_V02_E03_SumaEnR2(MiEscenaConVoz):
2      def construct(self):
3          self.set_speech_service( AzureService(voice="es-ES-AlvaroNeural" ) )
4          #self.set_speech_service(GTTSService(lang="es", tld="com"))
5          myTemplate = TextTemplate()
6          myTemplate.add_to_preamble(r"""\usepackage{nacal} """)
7
8          self.creditos()
9
10         axes = NumberPlane(x_range=(-4.5, 6.5, 1),
11                             y_range=(-1.5, 6.5, 1),
12                             background_line_style={
13                                 "stroke_width": 3,
14                                 "stroke_opacity": 0.4 }
15                             ).add_coordinates()
16
17
18         item0 = MathTex(r"\elemRp*{\Vect{a}+\Vect{b}}{i} = \eleVR{a}{i} + \eleVR{b}{i}",tex_template = myTemplate)
19         item1 = MathTex(r"\Vect{a}+\Vect{x} = \Vect{x}+\Vect{a}",tex_template = myTemplate).next_to(item0, DOWN, buff=0.5)
20         props_suma = VGroup(item0,item1).scale(1.5)
21
22     with self.voiceover(text=r""""Que la operación suma sea una
23 operación componente a componente""") as tracker:
24         self.play(FadeIn(props_suma[0]),
25                 run_time=tracker.duration)
26
27     with self.voiceover(text=r""""y que sea conmutativa""") as tracker:
28         self.play(Write(props_suma[1], run_time=tracker.duration/5))
29
30     with self.voiceover(text=r""""dota a la suma de interpretación
31 geométrica tanto en R 2 como en R 3.""") as tracker:
32         self.pausa(tracker.duration)
33         self.play(FadeOut(props_suma))
34
35         x = VectorR2([4,5], color=GREEN_B)
36         x_dot = x.dot(axes, radio=0.12)
37         x_tex = x.tex.scale(1.4)
38         vgr_x = VGroup(x.tex).next_to(x_dot, RIGHT).shift(RIGHT*.1)
39         x_v_line = x.v_line(axes)
40         x_h_line = x.h_line(axes)
41         with self.voiceover(text=r"""" Para verlo debemos interpretar
42 los vectores como puntos en el espacio, de manera que las
43 componentes de cada vector sean las coordenadas de un
44 punto.""") as tracker:
45             self.play(Create(axes), run_time=tracker.duration/2)
46
47         with self.voiceover(text=r""""En R 2, el convenio es considerar
48 que la primera componente es la coordenada respecto al eje
49 horizontal""") as tracker:
50             self.add(vgr_x)
51             self.pausa(tracker.duration/3)
52             self.play(Circumscribe(x_tex[0][1]), Indicate(x_v_line), run_time=tracker.duration*2/3)
53
54     with self.voiceover(text=r""""y la segunda como la coordenada

```



```

55     respecto al eje vertical.""") as tracker:
56         self.play(Circumscribe(x_tex[0][2]), Indicate(x_h_line), run_time=tracker.duration/2)
57         self.add(x_dot)
58         self.play(Indicate(x_dot), run_time=tracker.duration/2)
59
60     with self.voiceover(text=r""""Consecuentemente, vectores
61 distintos corresponden a puntos distintos.""") as tracker:
62         self.pausa(tracker.duration)
63         self.play(FadeOut(vgr_x, x_dot, x_h_line, x_v_line))
64
65     a = VectorR2([0,0], rpr='colum', color=YELLOW)
66     a_dot = a.dot(axes, radio=0.12)
67     a_tex = a.tex
68     vgr_a = VGroup(a.tex).next_to(a_dot, DOWN).shift(LEFT*.5)
69     with self.voiceover(text=r""""El vector cero corresponde con el
70 origen del sistema de coordenadas""") as tracker:
71         self.play(Indicate(a_dot), Indicate(a_tex), run_time=tracker.duration)
72         self.pausa
73
74     #añado punto en el eje horizontal quitando el anterior
75     b1 = VectorR2([3,0], rpr='colum')
76     b1_dot = b1.dot(axes, radio=0.12)
77     b1_diamond = Square(color=BLUE, fill_opacity=1, side_length=.12, fill_color=ORANGE).rotate(45*DEGREES).move_to(b1_dot)
78     b1_tex = b1.tex
79     vgr_b1= VGroup(b1_tex).next_to(b1_dot, DOWN)
80
81     with self.voiceover(text=r""""La primera componente de un
82 vector indica su coordenada respecto al eje horizontal. Los
83 valores positivos corresponden a posiciones a la derecha del
84 origen de coordenadas.""") as tracker:
85         self.play(FadeOut(a_dot, a_tex), FadeIn(b1_diamond))
86
87     #lo muevo y le pongo etiqueta
88     b1n = VectorR2([-2.5,0])
89     vgr_b1n= VGroup(b1n.tex).next_to(b1n.dot(axes, radio=0.12), DOWN)
90     b1n_diamond = Square(color=BLUE, fill_opacity=1, side_length=.12, fill_color=ORANGE).rotate(45*DEGREES).move_to(b1n.dot(axes, radio=0.12))
91     with self.voiceover(text=r""""Y valores negativos a posiciones a la
92 izquierda. Así, el vector 3 0 corresponde al punto del eje
93 horizontal que está 3 unidades a la derecha del origen.""") as tracker:
94         self.play(ReplacementTransform(b1_diamond, b1n_diamond), rate_function=exponential_decay, run_time=tracker.duration)
95         self.play(ReplacementTransform(b1n_diamond, b1_dot), FadeIn(b1.tex), rate_function=smooth, run_time=2*tracker.duration)
96
97     #añado punto inicial en el eje vertical
98     b2i = VectorR2([0,4])
99     vgr_b2i= VGroup(b2i.tex).next_to(b2i.dot(axes, radio=0.12), LEFT)
100     #b2i_dot = b2i.dot(axes, radio=0.12)
101     b2i_diamond = Square(color=BLUE, fill_opacity=1, side_length=.12, fill_color=ORANGE).rotate(45*DEGREES).move_to(b2i.dot(axes, radio=0.12))
102     with self.voiceover(text=r""""La segunda componente indica la
103 coordenada respecto al eje vertical. Valores positivos
104 corresponden a posiciones por encima del origen de
105 coordenadas. """) as tracker:
106         self.add(b2i_diamond)
107         self.pausa
108
109     # punto con oordenada negativa
110     b2n = VectorR2([0,-1])
111     vgr_b2n= VGroup(b2n.tex).next_to(b2n.dot(axes, radio=0.12), DOWN)
112     #b2n_dot = b2n.dot(axes, radio=0.12)
113     b2n_diamond = Square(color=BLUE, fill_opacity=1, side_length=.12, fill_color=ORANGE).rotate(45*DEGREES).move_to(b2n.dot(axes, radio=0.12))
114
115     #lo muevo y pongo etiqueta
116     b2 = VectorR2([0,2], rpr='colum')
117     b2_dot = b2.dot(axes, radio=0.12)
118     b2_tex = b2.tex
119     vgr_b2= VGroup(b2_tex).next_to(b2_dot, LEFT)
120     with self.voiceover(text=r""""Y valores negativos a posiciones
121 por debajo. Por tanto el vector 0 2 corresponde al punto del
122 eje vertical que está 2 unidades por encima del origen.""") as tracker:

```

```

123         self.play(ReplacementTransform(b2i_diamond, b2n_diamond, rate_function=exponential_decay, run_time= tracker.duration)
124         self.play(ReplacementTransform(b2n_diamond, b2_dot, rate_function=smooth, run_time=2*tracker.duration/3))
125         self.add(b2_tex)
126         self.pausa(n=3)
127
128         b = VectorR2([3,2], color=TEAL_A)
129         b_dot = b.dot(axes, radio=0.12)
130         b_tex = b.tex
131         vgr_b = VGroup(b_tex).next_to(b_dot, RIGHT)
132
133         # arriba añadir (0,3)+(1,0) = (3,1)
134         suma1_gr = VGroup(VectorR2([3,0]).tex,
135                           MathTex(r"+"),
136                           VectorR2([0,2]).tex,
137                           MathTex(r="="),
138                           b.tex.copy(),
139                           ).arrange(RIGHT).to_corner(UL)
140
141         with self.voiceover(text=r""Ahora consideremos la suma de
142         estos dos vectores. Se realiza componente a componente.""") as tracker:
143             self.add(suma1_gr[:3])
144             #self.pausa(3*tracker.duration/4)
145             #self.play(FadeIn(suma1_gr[3:]), run_time=tracker.duration/4)
146             #self.pausa_larga
147
148         with self.voiceover(text=r""Por una parte se suman las
149         coordenadas respecto al eje horizontal, y por otra las
150         coordenadas correspondientes al eje vertical. Así, el vector
151         suma es el vector 3 2.""") as tracker:
152             self.play(Circumscribe(suma1_gr[0][0][1]),
153                       Circumscribe(suma1_gr[2][0][1]),
154                       run_time=tracker.duration/3
155                       )
156             self.play(Circumscribe(suma1_gr[0][0][2]),
157                       Circumscribe(suma1_gr[2][0][2]),
158                       run_time=tracker.duration/3
159                       )
160             self.play(FadeIn(suma1_gr[3:]), run_time=tracker.duration/3)
161             self.pausa_larga
162
163         # pintar b con un punto y ejes y etiqueta
164         b_v_line = b.v_line(axes)
165         b_h_line = b.h_line(axes)
166         with self.voiceover(text=r""Sus componentes nos indican que
167         el punto está tres unidades a la derecha del origen y dos
168         unidades por encima.""") as tracker:
169             self.play(FadeIn(b_dot, b_tex, b_v_line, b_h_line))
170             self.pausa
171
172         # Añadir flechas ejes (quitando puntos) y desplazar para mostrar suma
173         flechab1 = b1.arrow(axes)
174         flechab2 = b2.arrow(axes)
175         with self.voiceover(text=r""Señalando la posición de cada
176         sumando con una flecha, podemos interpretar dicha flecha como
177         una indicación para llegar al punto.""") as tracker:
178             self.play(GrowArrow(flechab1),
179                       FadeOut(b1_dot),
180                       GrowArrow(flechab2),
181                       FadeOut(b2_dot),
182                       FadeOut(b_dot) )
183
184         with self.voiceover(text=r""Por ejemplo, al primer sumando se
185         llega desplazandose desde el origen tres unidades a la
186         derecha. De este modo dotamos a la suma de interpretación
187         geométrica.""") as tracker:
188             self.play(Indicate(b1_tex),
189                       run_time=tracker.duration/2)
190

```

```

191     # SUMA b1 + b2
192     a_dot_copy = a_dot.copy()
193     b1_dot_copy = b1_dot.copy()
194     b_dot_copy = b_dot.copy()
195     with self.voiceover(text=r""Sumar el primer vector con el
196     segundo corresponde a seguir las indicaciones del primer
197     vector""") as tracker:
198         self.play(#Indicate(flechab1),
199                 Indicate(b1_tex),
200                 Indicate(sumal_gr[0]),
201                 ReplacementTransform(a_dot_copy, b1_dot_copy),
202                 run_time=tracker.duration)
203
204     with self.voiceover(text=r""y luego seguir las indicaciones
205     del segundo.""") as tracker:
206         self.play(Indicate(b2_tex),
207                 Indicate(sumal_gr[2]),
208                 #Wiggle(flechab2),
209                 ReplacementTransform(b1_dot_copy, b_dot_copy),
210                 run_time=tracker.duration)
211
212     self.play(FadeOut(b_dot_copy))
213
214     # SUMA b2 + b1
215     a_dot_copy = a_dot.copy()
216     b2_dot_copy = b2_dot.copy()
217     b_dot_copy = b_dot.copy()
218     with self.voiceover(text=r""Pero invertir el orden y seguir
219     primero las indicaciones del segundo vector""") as tracker:
220         self.play(#Wiggle(flechab2),
221                 Indicate(b2_tex),
222                 Indicate(sumal_gr[2]),
223                 ReplacementTransform(a_dot_copy, b2_dot_copy),
224                 run_time=tracker.duration)
225
226     flechab = b.arrow(axes)
227     with self.voiceover(text=r""y después las indicaciones del
228     primero, nos conduce al mismo vector suma.""") as tracker:
229         self.play(Indicate(b1_tex),
230                 Indicate(sumal_gr[0]),
231                 #Wiggle(flechab1),
232                 ReplacementTransform(b2_dot_copy, b_dot_copy),
233                 run_time=tracker.duration/2)
234         self.play(GrowArrow(flechab),
235                 FadeOut(b_dot_copy),
236                 FadeOut(flechab1, b1_tex),
237                 FadeOut(flechab2, b2_tex),
238                 run_time=tracker.duration/2)
239
240     self.pausa
241     self.play(FadeOut(flechab), FadeIn(b_dot))
242     self.pausa_media
243
244     # arriba añadir (3,2)+(-2,1) = (1,3)
245     c = VectorR2([-2,1], color=PURPLE_A)
246     c_dot = c.dot(axes, radio=0.12)
247     c_tex = c.tex
248     c_v_line = c.v_line(axes)
249     c_h_line = c.h_line(axes)
250     vgr_c = VGroup(c.tex).next_to(c_dot(axes, radio=0.12), LEFT)
251
252     d = VectorR2([1,3], color=YELLOW_A)
253     d_dot = d.dot(axes, radio=0.12)
254     d_tex = d.tex
255     d_v_line = d.v_line(axes)
256     d_h_line = d.h_line(axes)
257     vgr_d = VGroup(d.tex).next_to(d_dot(axes, radio=0.12), UP)
258

```

```

259 suma2_gr = VGroup(b.tex.copy(),
260                    MathTex(r"+"),
261                    c.tex.copy(),
262                    MathTex(r "="),
263                    d.tex.copy(),
264                    ).arrange(RIGHT).to_corner(UL)
265
266 with self.voiceover(text=r""""Veamos otro ejemplo.""") as tracker:
267     self.play(FadeOut(suma1_gr),
268               run_time=tracker.duration )
269     self.pausa_corta
270
271 with self.voiceover(text=r""""Sumemos el último vector con el
272 vector -2 1.""") as tracker:
273     self.play(FadeIn(c_dot, c_tex, c_v_line, c_h_line))
274
275 with self.voiceover(text=r""""La suma de ambos es el vector 1
276 3.""") as tracker:
277     self.add(suma2_gr)
278     self.pausa
279     self.play(FadeOut(b_h_line, b_v_line, c_h_line, c_v_line))
280     self.add(d_dot, d_tex, d_v_line, d_h_line)
281     self.pausa_larga
282
283 # Añadir flechas ejes (quitando puntos) y desplazar para mostrar suma
284 flechab = b.arrow(axes)
285 flechac = c.arrow(axes)
286 flechad = d.arrow(axes)
287
288 with self.voiceover(text=r""""Una vez más, señalemos los
289 vectores con flechas.""") as tracker:
290     self.play(FadeOut(d_dot), #d_h_line, d_v_line),
291               GrowArrow(flechab),
292               FadeOut(b_dot),
293               GrowArrow(flechac),
294               FadeOut(c_dot))
295     self.pausa_corta
296
297 line_graph_b = axes.plot_line_graph(
298     x_values = [-2, 1],
299     y_values = [1, 3],
300     line_color=TEAL_E,
301     add_vertex_dots=False,
302     stroke_width = 3,
303 )
304
305 line_graph_c = axes.plot_line_graph(
306     x_values = [3, 1],
307     y_values = [2, 3],
308     line_color=PURPLE_E,
309     add_vertex_dots=False,
310     stroke_width = 3,
311 )
312
313 self.add(line_graph_b, line_graph_c)
314 a_dot_copy = a_dot.copy()
315 b_dot_copy = b_dot.copy()
316 d_dot_copy = d_dot.copy()
317
318 a1_diamond = Square(color=BLUE, fill_opacity=1, side_length=.12, fill_color=ORANGE).rotate(45*DEGREES).move_to(a_dot)
319 a2_diamond = Square(color=BLUE, fill_opacity=1, side_length=.12, fill_color=ORANGE).rotate(45*DEGREES).move_to(a_dot)
320 a1_diamond_copy = a1_diamond.copy()
321 a2_diamond_copy = a2_diamond.copy()
322 b1_diamond = Square(color=BLUE, fill_opacity=1, side_length=.12, fill_color=ORANGE).rotate(45*DEGREES).move_to(b1_dot)
323 b2_diamond = Square(color=BLUE, fill_opacity=1, side_length=.12, fill_color=ORANGE).rotate(45*DEGREES).move_to(b2_dot)
324 c1_diamond = Square(color=BLUE, fill_opacity=1, side_length=.12, fill_color=ORANGE).rotate(45*DEGREES).move_to(Dot(axes.
325 c2_diamond = Square(color=BLUE, fill_opacity=1, side_length=.12, fill_color=ORANGE).rotate(45*DEGREES).move_to(Dot(axes.
326 d1_diamond = Square(color=BLUE, fill_opacity=1, side_length=.12, fill_color=ORANGE).rotate(45*DEGREES).move_to(Dot(axes.

```

```

327     d2_diamond = Square(color=BLUE, fill_opacity=1, side_length=.12, fill_color=ORANGE).rotate(45*DEGREES).move_to(Dot(axes.
328
329
330     with self.voiceover(text=r""De nuevo, sumar el primer vector
331     con el segundo corresponde a seguir las indicaciones del
332     primer vector""") as tracker:
333         self.play(Indicate(suma2_gr[0]),
334                   ReplacementTransform(a_dot_copy, b_dot_copy),
335                   ReplacementTransform(a1_diamond, b1_diamond),
336                   ReplacementTransform(a2_diamond, b2_diamond),
337                   GrowArrow(flechab1),
338                   GrowArrow(flechab2),
339                   run_time=3)
340     self.play(FadeOut(flechab1,
341                      flechab2))
342
343     c1 = VectorR2([-2,0])
344     c2 = VectorR2([0,1])
345     flechac1 = c1.arrow(axes)
346     flechac2 = c2.arrow(axes)
347     flechac1d = flechac1.copy().move_to(axes.c2p(2,2,0))
348     flechac2d = flechac2.copy().move_to(axes.c2p(3,2.5,0))
349
350     with self.voiceover(text=r""y luego seguir las indicaciones
351     del segundo.""") as tracker:
352         self.play(Indicate(suma2_gr[2]),
353                   ReplacementTransform(b_dot_copy, d_dot_copy),
354                   ReplacementTransform(b1_diamond, d1_diamond),
355                   ReplacementTransform(b2_diamond, d2_diamond),
356                   GrowArrow(flechac1d),
357                   GrowArrow(flechac2d),
358                   run_time=3)
359     self.play(FadeOut(d_dot_copy,
360                      d1_diamond,
361                      d2_diamond,
362                      flechac1d,
363                      flechac2d))
364
365     a_dot_copy = a_dot.copy()
366     c_dot_copy = c_dot.copy()
367     d_dot_copy = d_dot.copy()
368     flechab1d = flechab1.copy().move_to(axes.c2p(-0.5,1,0))
369     flechab2d = flechab2.copy().move_to(axes.c2p(- 2,2,0))
370     with self.voiceover(text=r""Pero invertir el orden y seguir
371     primero las indicaciones del segundo vector""") as tracker:
372         self.play(Indicate(suma2_gr[2]),
373                   ReplacementTransform(a_dot_copy, c_dot_copy),
374                   ReplacementTransform(a1_diamond_copy, c1_diamond),
375                   ReplacementTransform(a2_diamond_copy, c2_diamond),
376                   GrowArrow(flechac1),
377                   GrowArrow(flechac2),
378                   run_time=3)
379     self.play(FadeOut(flechac1,
380                      flechac2))
381
382     with self.voiceover(text=r""y después las indicaciones del
383     primero, nos conduce al mismo punto.""") as tracker:
384         self.play(Indicate(suma2_gr[0]),
385                   ReplacementTransform(c_dot_copy, d_dot_copy),
386                   ReplacementTransform(c1_diamond, d1_diamond),
387                   ReplacementTransform(c2_diamond, d2_diamond),
388                   GrowArrow(flechab1d),
389                   GrowArrow(flechab2d),
390                   run_time=2*tracker.duration/3)
391     self.play(FadeOut(d_dot_copy),
392              FadeOut(d1_diamond),
393              FadeOut(d2_diamond),
394              FadeOut(d_v_line),

```

```

395         FadeOut(d_h_line),
396         FadeOut(flechab1d,
397                 flechab2d))
398     self.play(GrowArrow(flechad),
399               run_time=tracker.duration/3)
400     self.pausa_larga
401
402
403     with self.voiceover(text=r"""Esta descripción geométrica de la
404     suma, donde los sumandos forman un vértice de un
405     paralelogramo, y su suma es la diagonal que parte de dicho
406     vértice se denomina "regla del paralelogramo"."") as tracker:
407         self.play(Indicate(flechab),
408                   Indicate(flechac),
409                   run_time=tracker.duration/2 )
410         self.play(Indicate(flechad),
411                   run_time=tracker.duration/2 )
412
413     with self.voiceover(text=r"""A pesar de la utilidad de las
414     flechas, recuerde que un vector es una lista de números, y que
415     podemos hacer corresponder dichos números con las coordenadas
416     de un punto en el espacio. Por ello, la representación
417     geométrica del vector es el punto. La flecha tan solo lo
418     señala.""") as tracker:
419         self.play(Indicate(b_tex),
420                   Indicate(c_tex),
421                   Indicate(d_tex),
422                   run_time=tracker.duration/2 )
423         self.play(FadeOut(flechab,
424                             flechac,
425                             flechad,
426                             line_graph_b,
427                             line_graph_c),
428                   FadeIn(b_dot, c_dot, d_dot),
429                   run_time=tracker.duration/2 )
430
431     b_dot_copia=Dot(axes.c2p(*b.coords), radius=0.01)
432     c_dot_copia=Dot(axes.c2p(*c.coords), radius=0.01)
433     d_dot_copia=Dot(axes.c2p(*d.coords), radius=0.01)
434     with self.voiceover(text=r"""Una de las dificultades para
435     representar los puntos es que su dimensión es cero.""") as tracker:
436         self.play(
437             Transform(b_dot, b_dot_copia),
438             Transform(c_dot, c_dot_copia),
439             Transform(d_dot, d_dot_copia),
440             run_time = 6*tracker.duration/5 )
441
442     with self.voiceover(text=r"""Una solución es indicar para
443     cada punto su coordenada en el eje horizontal (es decir, el
444     primer número de la lista).""") as tracker:
445         self.play(FadeIn(b_v_line),
446                   FadeIn(c_v_line),
447                   FadeIn(d_v_line),
448                   run_time = tracker.duration/2)
449         self.play(Circumscribe(b_tex[0][1]),
450                   Circumscribe(c_tex[0][1:3]),
451                   Circumscribe(d_tex[0][1]),
452                   run_time = tracker.duration/2)
453
454     with self.voiceover(text=r"""y su coordenada en el eje
455     vertical (es decir, el segundo número de la lista).""") as tracker:
456         self.play(FadeIn(b_h_line),
457                   FadeIn(c_h_line),
458                   FadeIn(d_h_line),
459                   run_time = tracker.duration/2)
460         self.play(Circumscribe(b_tex[0][2]),
461                   Circumscribe(c_tex[0][3]),
462                   Circumscribe(d_tex[0][2]),

```

```

463         run_time = tracker.duration/2)
464
465     with self.voiceover(text=r""""Sin embargo, la representación
466     más frecuente son las flechas. Se ven bien y arrojan una
467     interpretación intuitiva de la suma de vectores. """) as tracker:
468         self.play(FadeIn(flechab,flechac,fle Chad),
469                 FadeOut(b_h_line, b_v_line),
470                 FadeOut(c_h_line, c_v_line),
471                 FadeOut(d_h_line, d_v_line),
472                 run_time = tracker.duration/2 )
473         self.play(FadeIn(line_graph_b, line_graph_c),
474                 run_time = tracker.duration/2 )
475
476     with self.voiceover(text=r""""Pero no debe olvidar que nuestra
477     definición de vector de  $\mathbb{R}^n$  es que es una lista de números. Y
478     que su representación geométrica hace corresponder dichos
479     números con las coordenadas de puntos en el espacio. Por
480     tanto, cuando veamos un vector representado con una flecha,
481     debemos recordar que el vector no es la flecha. El vector es
482     el punto señalado por la flecha.""") as tracker:
483         self.play(FadeOut(line_graph_b, line_graph_c),
484                 Indicate(b_tex),
485                 Indicate(c_tex),
486                 Indicate(d_tex),
487                 run_time=tracker.duration/2 )
488         self.play(FadeOut(flechab, flechac, fle Chad),
489                 FadeIn(b.dot(axes)),
490                 FadeIn(c.dot(axes)),
491                 FadeIn(d.dot(axes)),
492                 run_time=tracker.duration/2)
493
494     self.pausa_muy_larga

```

1.1.4. Escena 4 - Interpretación geométrica de la suma en \mathbb{R}^3

1. Escena 4(voz)

```

1  class L01_V02_E04_SumaEnR3_voz(MiEscenaConVoz):
2      def construct(self):
3          self.set_speech_service( AzureService(voice="es-ES-AlvaroNeural" ) )
4          #self.set_speech_service(GTTSService(lang="es", tld="com"))
5          myTemplate = TextTemplate()
6          myTemplate.add_to_preamble(r"""\usepackage{nacal} """)
7
8          # Portada
9          titulo = Title(r"Interpretación de la suma en  $\mathbb{R}^3$ ",
10                        tex_template = myTemplate,
11                        font_size=70).set_color(BLUE)
12
13          a = nc.Vector(sp.symbols('a:4')[1:])
14          b = nc.Vector(sp.symbols('b:4')[1:])
15          s1 = MathTex(a.latex(), tex_template = myTemplate,)
16          mas= MathTex(r"+", tex_template = myTemplate,)
17          s3 = MathTex(b.latex(), tex_template = myTemplate,)
18          igual = MathTex(r "=", tex_template = myTemplate,)
19          s5 = MathTex((a+b).latex(), tex_template = myTemplate,)
20          grp1 = VGroup(s1,mas,s3,igual,s5,igual.copy(),s3.copy(),mas.copy(),s1.copy()).arrange(RIGHT)
21
22          self.creditos(17)
23
24          with self.voiceover(text=r""""La representación geométrica en
25           $\mathbb{R}^3$  es similar. """) as tracker:
26              self.add(titulo)
27              self.play(FadeIn(grp1[0]))
28

```

```

29     with self.voiceover(text=r""""El convenio es interpretar las
30     dos primeras componentes como coordenadas respecto a un plano
31     horizontal""") as tracker:
32         self.play(Indicate(grp1[0][0][2:6]),
33                   run_time=tracker.duration)
34
35     with self.voiceover(text=r""""y la tercera como la coordenada respecto a un eje
36     perpendicular al plano.""") as tracker:
37         self.play(Indicate(grp1[0][0][6:8]),
38                   run_time=tracker.duration)
39
40     with self.voiceover(text=r""""De nuevo, como la suma se realiza
41     componente a componente y es conmutativa""") as tracker:
42         self.play(FadeIn(grp1[1:5]),
43                   run_time=tracker.duration/2)
44         self.play(FadeIn(grp1[5:]),
45                   run_time=tracker.duration/2)
46
47     with self.voiceover(text=r""""su representación geométrica en
48     R3 también verifica la regla del paralelogramo.""") as tracker:
49         self.pausa(tracker.duration)

```

2. Escena 4 (Visión 3D)

```

1  class L01_V02_E04_SumaEnR3_3D(ThreeDScene):
2      <<Créditos en distintas partes de la pantalla>>
3
4      def construct(self):
5          axes = ThreeDAxes()
6
7          x_label = axes.get_x_axis_label(Tex("1ª comp."))
8          y_label = axes.get_y_axis_label(Tex("2ª comp.")).shift(UP * 2.4).shift(LEFT * 0.6)
9
10
11         self.credits(17)
12
13         # zoom out so we see the axes
14         self.set_camera_orientation(zoom=0.5)
15
16         self.play(FadeIn(axes), FadeIn(x_label), FadeIn(y_label))
17
18         self.wait(1)
19
20         # animate the move of the camera to properly see the axes
21         self.move_camera(phi=75 * DEGREES, theta=60 * DEGREES, zoom=1, run_time=1.5)
22
23         # built-in updater which begins camera rotation
24         self.begin_ambient_camera_rotation(rate=0.2)
25
26         self.wait(2)
27
28
29         b = VectorR3([3,2,3], color=TEAL_A)
30         c = VectorR3([-2,1,1], color=PURPLE_A)
31         d = VectorR3([1,3,4], color=YELLOW_A)
32
33         b_dot = b.dot(axes)
34         c_dot = c.dot(axes)
35         d_dot = d.dot(axes)
36
37         line_x = Line3D(start=np.array(axes.c2p(3,0,0)), end=np.array(axes.c2p(3,2,0)), thickness=0.01)
38         line_y = Line3D(start=np.array(axes.c2p(0,2,0)), end=np.array(axes.c2p(3,2,0)), thickness=0.01)
39         line_z = Line3D(start=np.array(axes.c2p(3,2,0)), end=np.array(axes.c2p(3,2,3)), thickness=0.01)
40
41         flechab = b.arrow(axes)
42         flechac = c.arrow(axes)

```



```

43     flechad = d.arrow(axes)
44
45     linebd = Line3D(start=np.array(axes.c2p(*b.coords)), end=np.array(axes.c2p(*d.coords)), thickness=0.01)
46     linecd = Line3D(start=np.array(axes.c2p(*c.coords)), end=np.array(axes.c2p(*d.coords)), thickness=0.01)
47
48
49     self.play(FadeIn(line_x))
50     self.play(FadeIn(line_y))
51     self.wait(4)
52     self.play(FadeIn(line_z))
53     self.add(b_dot)
54
55     self.wait(1.5)
56
57     self.play(FadeIn(flechab),
58               FadeOut(b_dot),)
59
60     self.wait(1.5)
61
62     self.play(FadeIn(flechac))
63     self.play(FadeOut(line_x, line_y, line_z))
64
65     self.wait(1.5)
66
67     self.play(FadeIn(linebd),
68               FadeIn(linecd),)
69
70     self.wait(1.5)
71
72     self.play(FadeIn(flechad))
73
74     self.wait(2)
75
76     #self.play(FadeOut(flechab, flechac, flechad, linebd, linecd),
77     #          FadeIn(b_dot, c_dot, d_dot))
78
79     #self.wait(2)
80
81
82     #self.move_camera(phi=0 * DEGREES, theta=0 * DEGREES, zoom=1, run_time=1.5)
83
84     #self.wait(2)

```

■ Fusión audio y vídeo poca calidad

```

1  rm -f L01_V02_E04_SumaEnR3.mp4
2  ffmpeg -i L01_V02_E04_SumaEnR3_3D.mp4 -i L01_V02_E04_SumaEnR3_voz.mp4 -c:v copy -c:a aac -strict experimental L01_V02_E04_
3  #mv L01_V02_E04_SumaEnR3_voz.srt L01_V02_E04_SumaEnR3.srt
4  mkdir -p aux_movie_files
5  mv L01_V02_E04_SumaEnR3_3D.mp4 aux_movie_files/
6  mv L01_V02_E04_SumaEnR3_voz.mp4 aux_movie_files/

```

■ Fusión audio y vídeo poca calidad HD1080

```

1  rm -f L01_V02_E04_SumaEnR3.mp4
2  ffmpeg -i L01_V02_E04_SumaEnR3_3D.mp4 -i L01_V02_E04_SumaEnR3_voz.mp4 -c:v copy -c:a aac -strict experimental L01_V02_E04_
3  cp L01_V02_E04_SumaEnR3_voz.srt L01_V02_E04_SumaEnR3.srt
4  mkdir -p aux_movie_files
5  mv L01_V02_E04_SumaEnR3_3D.mp4 aux_movie_files/
6  mv L01_V02_E04_SumaEnR3_voz.mp4 aux_movie_files/

```

1.1.5. Escena 5 - Interpretación geométrica de la suma en \mathbb{R}^n

1. Escena 5 (voz)

```
1 class L01_V02_E05_SumaEnRn_voz(MiEscenaConVoz):
2     def construct(self):
3         self.set_speech_service( AzureService(voice="es-ES-AlvaroNeural" ) )
4         #self.set_speech_service(GTTSService(lang="es", tld="com"))
5         myTemplate = TexTemplate()
6         myTemplate.add_to_preamble(r"""\usepackage{nacal} """)
7
8         # Portada
9         titulo = Title(r"Interpretación de la suma en  $\mathbb{R}^n$ ",
10                        tex_template = myTemplate,
11                        font_size=70).set_color(BLUE)
12
13         a = nc.Vector(sp.symbols('a:4')[1:])
14         b = nc.Vector(sp.symbols('b:4')[1:])
15         s1 = MathTex(a.latex(), tex_template = myTemplate,)
16         mas= MathTex(r"+", tex_template = myTemplate,)
17         s3 = MathTex(b.latex(), tex_template = myTemplate,)
18         igual = MathTex(r "=", tex_template = myTemplate,)
19         s5 = MathTex((a+b).latex(), tex_template = myTemplate,)
20         grp1 = VGroup(s1,mas,s3,igual,s5,igual.copy(),s3.copy(),mas.copy(),s1.copy()).arrange(RIGHT)
21
22         self.creditos(3)
23
24         with self.voiceover(text=r""""Los vectores en  $\mathbb{R}^n$  son puntos en
25 un espacio ene-dimensional. Para representarlos sería
26 necesario dibujar tantos ejes de coordenadas como elementos
27 tiene el vector. Esto no es posible cuando el número de
28 componentes es mayor a tres.""") as tracker:
29             self.add(titulo)
30             self.play(FadeIn(grp1[0]))
31
32         with self.voiceover(text=r""""No obstante, sí que podemos
33 recurrir a una interpretación geométrica. Dicha interpretación
34 no describe literalmente las componentes de cada vector. Es
35 tan solo un ESQUEMA geométrico.""") as tracker:
36             self.play(Indicate(grp1[0][0][2:6]),
37                      run_time=tracker.duration)
38
39         with self.voiceover(text=r""""En dicho esquema, los vectores
40 son puntos de un espacio ene-dimensional. Como en los casos
41 anteriores, se suman componente a componente, es decir, se
42 suman las coordenadas respecto a cada eje de manera separada,
43 y su suma es conmutativa.""") as tracker:
44             self.play(Indicate(grp1[0][0][6:8]),
45                      run_time=tracker.duration)
46
47         with self.voiceover(text=r""""Por tanto, como esquema
48 geométrico, la regla del paralelogramo es válida incluso en
49 espacios de dimension arbitraria. Lo es incluso en dimensión
50 infinita.""") as tracker:
51             self.play(FadeIn(grp1[1:5]),
52                      run_time=tracker.duration/2)
53             self.play(FadeIn(grp1[5:]),
54                      run_time=tracker.duration/2)
```

2. Escena 5 (Visión 3D)

```
1 class L01_V02_E05_SumaEnRn_3D(ThreeDScene):
2     <<Créditos en distintas partes de la pantalla>>
3
4     def construct(self):
```

```

5     #self.set_speech_service( AzureService(voice="es-ES-AlvaroNeural" ) )
6     #self.set_speech_service(GTSService(lang="es", tld="com"))
7     myTemplate = TextTemplate()
8     myTemplate.add_to_preamble(r"""\usepackage{nacal} """)
9
10    #self.creditos(17)
11
12    #plane = NumberPlane(background_line_style={"stroke_opacity": 0.1})
13
14    axes = ThreeDAxes()
15
16    b      = VectorR3([2,2,3], color=PURE_RED)
17    c      = VectorR3([-3,1,-1], color=PURE_GREEN)
18    d      = VectorR3([-1,3,2], color=PURE_BLUE)
19
20    b_dot = b.dot(axes)
21    c_dot = c.dot(axes)
22    d_dot = d.dot(axes)
23
24    flechab = b.arrow(axes)
25    flechac = c.arrow(axes)
26    flechad = d.arrow(axes)
27
28    linebd = Line3D(start=np.array(axes.c2p(*b.coords)), end=np.array(axes.c2p(*d.coords)))
29    linecd = Line3D(start=np.array(axes.c2p(*c.coords)), end=np.array(axes.c2p(*d.coords)))
30
31    #self.add(axes, plane)
32
33    self.move_camera(phi=75 * DEGREES, theta=60 * DEGREES, zoom=1, run_time=1)
34    self.add(b_dot,
35             c_dot)
36    self.begin_ambient_camera_rotation(rate=0.2)
37    self.wait(15)
38
39    self.play(FadeIn(flechab,
40                    flechac),
41             FadeOut(b_dot,
42                    c_dot))
43    self.wait(23)
44
45    self.add(linebd,
46            linecd)
47    self.play(FadeIn(flechad))
48
49    self.wait(3)
50
51    self.begin_ambient_camera_rotation(rate=0.6, about='gamma')
52    self.wait(5)
53
54    self.begin_ambient_camera_rotation(rate=0.6, about='theta')
55    self.play(FadeIn(b_dot,
56                    c_dot,
57                    d_dot),
58             FadeOut(flechab,
59                    flechac,
60                    flechad,
61                    linebd,
62                    linecd))
63    self.wait(5)

```

■ Fusión audio y vídeo poca calidad

```

1  rm -f L01_V02_E05_SumaEnRn.mp4
2  ffmpeg -i L01_V02_E05_SumaEnRn_3D.mp4 -i L01_V02_E05_SumaEnRn_voz.mp4 -c:v copy -c:a aac -strict experimental L01_V02_E05
3  #mv L01_V02_E05_SumaEnRn_voz.srt L01_V02_E05_SumaEnRn.srt
4  mkdir -p aux_movie_files

```

```

5 mv L01_V02_E05_SumaEnRn_3D.mp4 aux_movie_files/
6 mv L01_V02_E05_SumaEnRn_voz.mp4 aux_movie_files/

```

- Fusión audio y vídeo poca calidad HD1080

```

1 rm -f L01_V02_E05_SumaEnRn.mp4
2 ffmpeg -i L01_V02_E05_SumaEnRn_3D.mp4 -i L01_V02_E05_SumaEnRn_voz.mp4 -c:v copy -c:a aac -strict experimental L01_V02_E05_SumaEnRn.mp4
3 mkdir -p aux_movie_files
4 cp L01_V02_E05_SumaEnRn_voz.srt L01_V02_E05_SumaEnRn.srt
5 mv L01_V02_E05_SumaEnRn_3D.mp4 aux_movie_files/
6 mv L01_V02_E05_SumaEnRn_voz.mp4 aux_movie_files/

```

1.1.6. Escena 6 - Resumen

```

1 class L01_V02_E06_Resumen(MiEscenaConVoz):
2     def construct(self):
3         self.set_speech_service( AzureService(voice="es-ES-AlvaroNeural" ) )
4         #self.set_speech_service(GTTSService(lang="es", tld="com"))
5         myTemplate = TextTemplate()
6         myTemplate.add_to_preamble(r"""\usepackage{nacal} """)
7
8         self.creditos()
9
10        titulo = Title(r"Suma de vectores de  $\mathbb{R}^n$ ",
11                        tex_template = myTemplate,
12                        font_size=70).set_color(BLUE)
13        self.play(Write(titulo))
14        self.pausa()
15
16        # Resumen
17        resumen = Tex(r"\textbf{Lo más importante:}",
18                     tex_template = myTemplate,
19                     font_size=50).set_color(ORANGE).next_to(titulo, DOWN, aligned_edge=LEFT)
20
21
22        with self.voiceover(text=r"""\textbf{Por último, quiero subrayar que la interpretación geométrica se deriva de la definición de la suma.}""") as tracker:
23            self.add(resumen)
24            self.pausa(tracker.duration)
25
26
27        cvab = MathTex(r"\textbf{el elemento de } \mathbb{V} \text{ es } \mathbb{V}^n", tex_template = myTemplate)
28        cva = MathTex(r"\textbf{el elemento de } \mathbb{V}^n", tex_template = myTemplate)
29        cvb = MathTex(r"\textbf{el elemento de } \mathbb{V}^n", tex_template = myTemplate)
30        igual = MathTex(r"=", tex_template = myTemplate,)
31        mas = MathTex(r"+", tex_template = myTemplate,)
32        eq_suma = VGroup(cvab,igual,cva,mas,cvab).arrange(RIGHT).scale(1.5)
33        cva_copy = cva.copy().move_to(cvb)
34        cvb_copy = cvb.copy().move_to(cva)
35
36
37        item1 = MathTex(r"\textbf{\mathbb{V}^n + \mathbb{V}^n = \mathbb{V}^n + \mathbb{V}^n",tex_template = myTemplate).next_to(eq_suma, DOWN, buff=1.5).scale(1.5)
38
39        props_suma = VGroup(eq_suma, item1)
40
41        with self.voiceover(text=r"""\textbf{Por tanto, lo más importante es destacar que la definición indica que la suma es una operación componente a componente}""") as tracker:
42            self.play(FadeIn(props_suma[0]),
43                    run_time=tracker.duration+0.3)
44
45
46
47        with self.voiceover(text=r"""\textbf{Ello se traduce en una regla de cálculo simbólico. Dicha regla nos dice que el operador selector es distributivo respecto de la suma.}""") as tracker:
48
49

```

```

50         self.pausa(tracker.duration/2)
51         self.play(Indicate(eq_suma[0][0][0]),
52                   Indicate(eq_suma[0][0][-3:]),
53                   Indicate(eq_suma[2][0][-2:]),
54                   Indicate(eq_suma[4][0][-2:]),
55                   run_time = tracker.duration/2)
56         self.pausa(0.3)
57
58         with self.voiceover(text=r"""Además, como las componentes son
59 números reales, también hay que destacar que la suma es
60 conmutativa""") as tracker:
61             self.play(Transform(cva,cva_copy),
62                       Transform(cvb, cvb_copy),
63                       run_time = 3*tracker.duration/4)
64             self.play(Indicate(item1),
65                       run_time = 3*tracker.duration/10)
66
67         self.pausa_larga()
68

```

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, buff=0.1).set_color(GRAY_D)
3 self.add(CGG)
```

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

3. Rodando: 1,2,3... ¡acción!

1. Generamos un fichero mpeg por cada escena

- Versión de poca calidad

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

- Versión calidad HD1080

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

- Montando la versión de resolución HD1080

```
1 ln -s -f "$(pwd)/$subdir/ZCreditos/$calidad/ZCreditos.mp4" "$(pwd)/$subdir/$video/$calidad/ZCreditos.mp4"
2 rm -f $subdir/$video/$calidad/$video.mp4 list.txt
3 for f in $subdir/$video/$calidad/*.mp4 ; do echo file '$f' >> list.txt; done && ffmpeg -f concat -safe 0 -i list.txt -c
4
5 mkdir -p tmp
6
7 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 <<<
```

4. Copiamos el resultado a un lugar público

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