

---

# TRES EN RAYA CON RASPBERRY PI

---

DAVID ÁLVAREZ

GUILLERMO CREUS



*Escuela Técnica Superior de Ingeniería Industrial de Barcelona*  
*Universidad Politécnica de Cataluña*

BARCELONA, 1 DE MAYO DE 2019

# Índice

<b>1. Introducción</b>	<b>1</b>
1.1. Objetivos . . . . .	1
1.2. Alcance . . . . .	1
1.3. Antecedentes . . . . .	1
<b>2. Herramientas e implementación</b>	<b>1</b>
<b>3. Funcionamiento</b>	<b>1</b>
3.1. Tres en Raya . . . . .	1
3.2. Movimiento . . . . .	1
<b>4. Planificación y costes</b>	<b>1</b>
<b>5. Resultados y conclusiones</b>	<b>1</b>
<b>6. Bibliografía</b>	<b>1</b>
<b>A. Código completo</b>	<b>4</b>
A.1. Estrategia . . . . .	4
A.2. Movimiento . . . . .	14
A.3. Programa principal . . . . .	21
A.4. Animación . . . . .	26
A.5. Servidor . . . . .	40

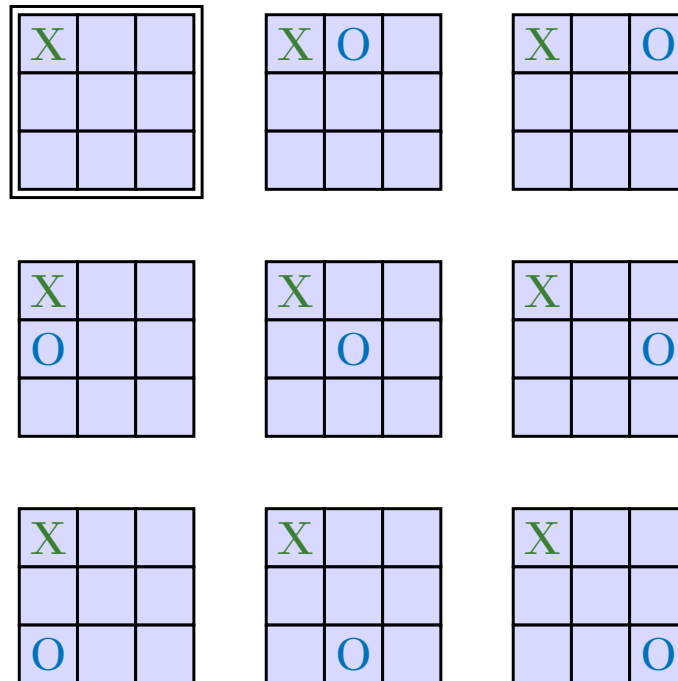


Figura 1: Tableros simétricos equivalentes.

## 1. Introducción

Este proyecto ha consistido en el desarrollo de un programa capaz de jugar de manera inteligente al juego del Tres en Raya.

### 1.1. Objetivos

### 1.2. Alcance

### 1.3. Antecedentes

## 2. Herramientas e implementación

## 3. Funcionamiento

### 3.1. Tres en Raya

### 3.2. Movimiento

## 4. Planificación y costes

## 5. Resultados y conclusiones

## 6. Bibliografía

- Tres en Raya

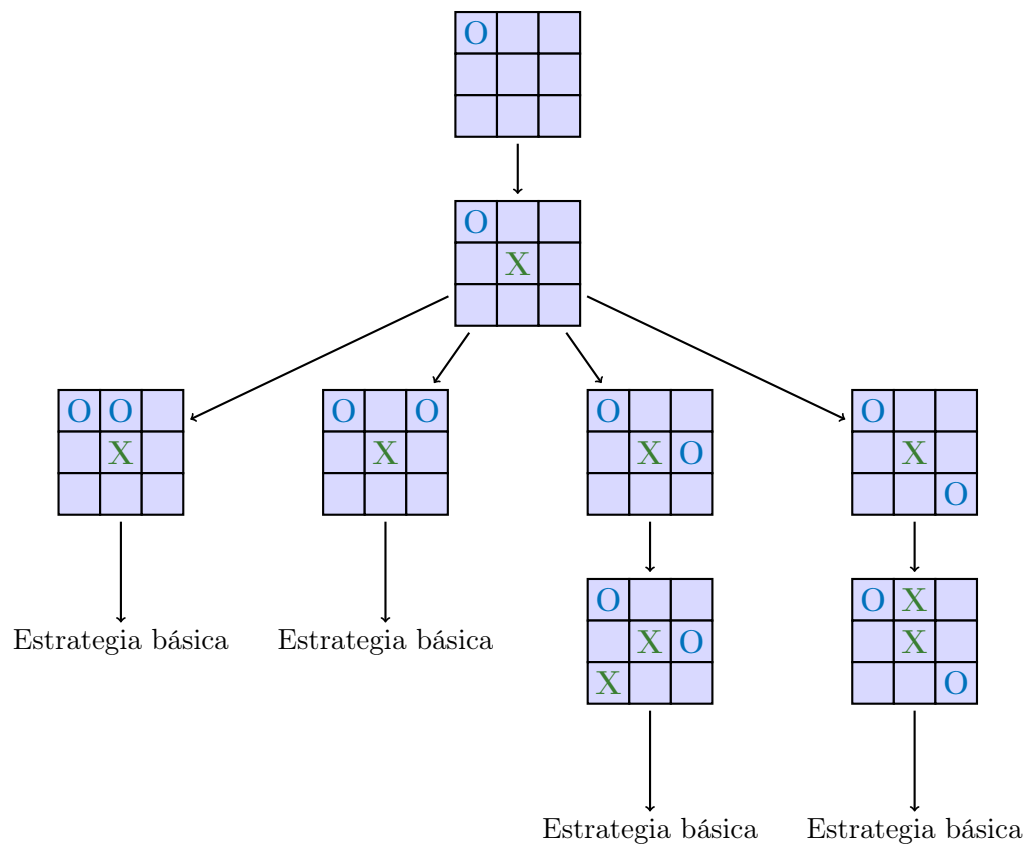


Figura 2: Una rama como ejemplo.

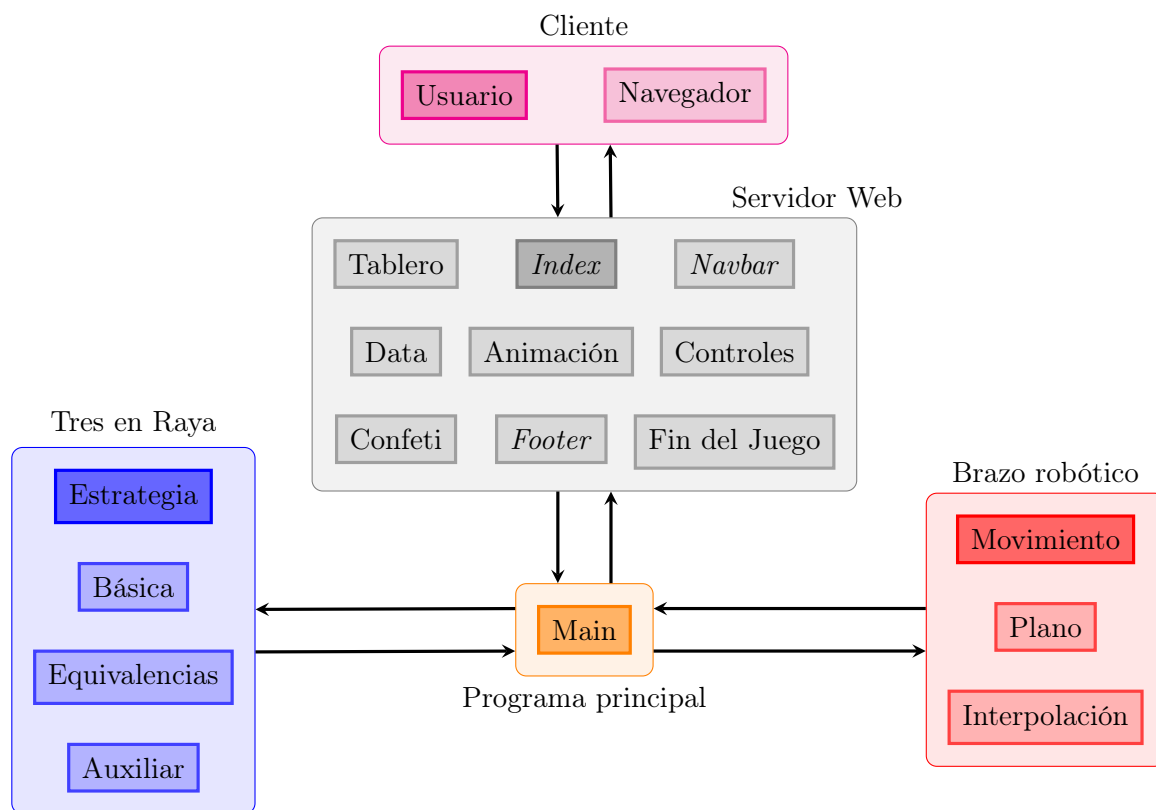


Figura 3: Arquitectura del programa.

- Estrategia: estrategia.py

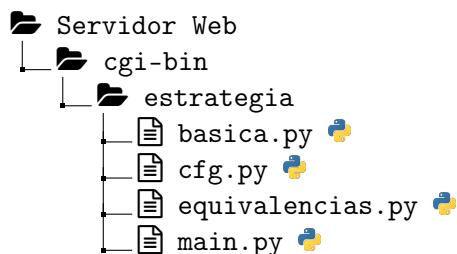
A continuación se muestra el árbol de Se recogen mostrando primero los directorios y después en orden alfabético.

## A. Código completo

A continuación se recoge todo el código desarrollado en este proyecto.

### A.1. Estrategia

Aquí se recoge todo el código desarrollado en Python 🐍 relacionado con la estrategia de juego del Tres en Raya. Este código en el servidor se encuentra (de acuerdo con la figura 4 de la página 5) en el siguiente directorio:



#### Estrategia Básica (*basica.py*)

```
basica.py
1  """
2  Estrategia de juego básica y movimiento aleatorio (usado para los niveles de
3  dificultad).
4  """
5
6  import sys
7  import random
8
9
10 def game_end(M):
11     x0=M[0][0]+M[1][0]+M[2][0]
12     x1=M[0][1]+M[1][1]+M[2][1]
13     x2=M[0][2]+M[1][2]+M[2][2]
14     y0=M[0][0]+M[0][1]+M[0][2]
15     y1=M[1][0]+M[1][1]+M[1][2]
16     y2=M[2][0]+M[2][1]+M[2][2]
17     d1=M[0][0]+M[1][1]+M[2][2]
18     d2=M[0][2]+M[1][1]+M[2][0]
19
20     if x0 == 0 or x1 == 0 or x2 == 0 or y0 == 0 or y1 == 0 or y2 == 0 \
21        or d1 == 0 or d2 == 0:
22         return "User wins"
23     if x0 == 3 or x1 == 3 or x2 == 3 or y0 == 3 or y1 == 3 or y2 == 3 \
24        or d1 == 3 or d2 == 3:
```

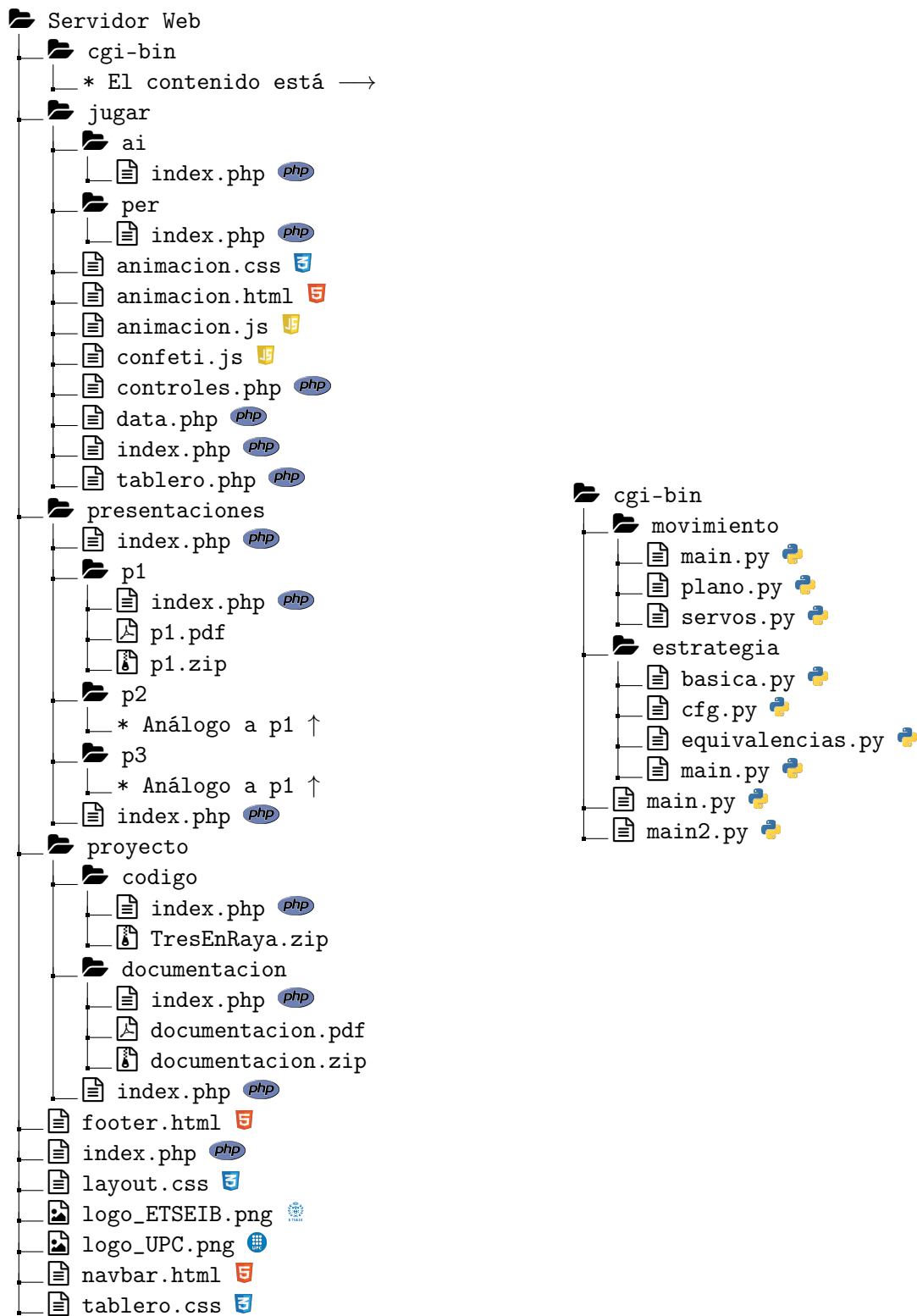


Figura 4: Árbol de directorios.

```

25         return "AI wins"
26
27     tie = True
28     for i in range(3):
29         for j in range(3):
30             if M[i][j] == -3:
31                 tie = False
32                 break
33
34     if tie:
35         return "Tie"
36
37     return "Not ended"
38
39
40 def check_win(M,i,j):
41     #Comprobamos posibles jugadas ganadoras
42     if (i==0 and j==0):
43         if (M[1][0]+M[2][0]==2 or M[0][1]+M[0][2]==2 or M[1][1]+M[2][2]==2):
44             ↪ return 1
45
46     elif (i==0 and j==1):
47         if (M[0][0]+M[0][2]==2 or M[1][1]+M[2][1]==2): return 1
48
49     elif (i==0 and j==2):
50         if (M[0][0]+M[0][1]==2 or M[1][2]+M[2][2]==2 or M[2][0]+M[1][1]==2):
51             ↪ return 1
52
53     elif (i==1 and j==0):
54         if (M[0][0]+M[2][0]==2 or M[1][1]+M[1][2]==2): return 1
55
56     elif (i==1 and j==1):
57         if (M[0][0]+M[2][2]==2 or M[2][0]+M[0][2]==2 or M[0][1]+M[2][1]==2 or
58             ↪ M[1][0]+M[1][2]==2): return 1
59
60     elif (i==1 and j==2):
61         if (M[1][0]+M[1][1]==2 or M[0][2]+M[2][2]==2): return 1
62
63     elif (i==2 and j==0):
64         if (M[2][1]+M[2][2]==2 or M[0][0]+M[1][0]==2 or M[1][1]+M[0][2]==2):
65             ↪ return 1
66
67     elif (i==2 and j==1):
68         if (M[2][0]+M[2][2]==2 or M[0][1]+M[1][1]==2): return 1
69
70     elif (i==2 and j==2):
71         if (M[2][0]+M[2][1]==2 or M[0][2]+M[1][2]==2 or M[0][0]+M[1][1]==2):
72             ↪ return 1
73
74     return 0
75
76 def check(M,i,j):
77     #Comprobamos posibles jaques

```



```

73     if (i==0 and j==0):
74         if (M[1][0]+M[2][0]==0 or M[0][1]+M[0][2]==0 or M[1][1]+M[2][2]==0):
75             ↪ return 1
76
77     elif (i==0 and j==1):
78         if (M[0][0]+M[0][2]==0 or M[1][1]+M[2][1]==0): return 1
79
80     elif (i==0 and j==2):
81         if (M[0][0]+M[0][1]==0 or M[1][2]+M[2][2]==0 or M[2][0]+M[1][1]==0):
82             ↪ return 1
83
84     elif (i==1 and j==0):
85         if (M[0][0]+M[2][0]==0 or M[1][1]+M[1][2]==0): return 1
86
87     elif (i==1 and j==1):
88         if (M[0][0]+M[2][2]==0 or M[2][0]+M[0][2]==0 or M[0][1]+M[2][1]==0 or
89             ↪ M[1][0]+M[1][2]==0): return 1
90
91     elif (i==1 and j==2):
92         if (M[1][0]+M[1][1]==0 or M[0][2]+M[2][2]==0): return 1
93
94     elif (i==2 and j==0):
95         if (M[2][1]+M[2][2]==0 or M[0][0]+M[1][0]==0 or M[1][1]+M[0][2]==0):
96             ↪ return 1
97
98     elif (i==2 and j==1):
99         if (M[2][0]+M[2][2]==0 or M[0][1]+M[1][1]==0): return 1
100
101     elif (i==2 and j==2):
102         if (M[2][0]+M[2][1]==0 or M[0][2]+M[1][2]==0 or M[0][0]+M[1][1]==0):
103             ↪ return 1
104
105     return 0
106
107 def moveRandom(M):
108     """
109     Mover aleatoriamente a una casilla vacía.
110     """
111     movs = []
112     for ip in range(3):
113         for jp in range(3):
114             if M[ip][jp] == -3:
115                 movs.append([ip, jp])
116     if len(movs) > 0:
117         [ip, jp] = movs[random.randint(0, len(movs) - 1)]
118         M[ip][jp] = 1
119         return ip, jp
120     return -1, -1
121
122 def move(M):
123     #Entrada de matriz M 3x3 con 0s (humano), 1s (máquina) y previamente
124     ↪ inicializada en "-3"s (importantente que sea así para que funcione -- por
125     ↪ tema sumas de check_win) --> decide una jugada para la máquina (estrategia
126     ↪ basic)

```

```

121
122     #Compruebo jugadas ganadoras para la máquina
123     for i in range(3):
124         for j in range(3):
125             if(M[i][j] == -3 and check_win(M,i,j)):
126                 return i,j
127
128     for i in range(3):
129         for j in range(3):
130             if(M[i][j]==-3 and check(M,i,j)):
131                 return i,j
132     return -1,-1
133
134
135 def moveBasic(M):
136     i, j = move(M)
137     # Mover al azar si no hay ningún movimiento.
138     if i == -1 and j == -1:
139         return moveRandom(M)
140
141     M[i][j] = 1
142     return i, j

```

basica.py

### Auxiliar (cfg.py)

```

1     """
2     Usado únicamente para poder acceder a las variables de manera global entre
3     módulos.
4     """
5
6     # Árbol de decisiones: 3 ramas con los nodos en orden y 3 vectores de conexiones
7     # entre nodos.
8     rama1 = [
9         [0, 0],
10        [1, 1],
11        [0, 1], [0, 2], [1, 2], [2, 2],
12        "EB", "EB", [2, 1], [0, 1],
13        "EB", "EB"
14    ]
15    conex1 = [
16        [1], [2, 3, 4, 5], [6], [7], [8], [9], [], [], [10], [11], [], []
17    ]
18    rama2 = [
19        [1, 1],
20        [0, 0],
21        [0, 1], [0, 2], [1, 2], [2, 2],
22        "EB", "EB", "EB", [0, 2],
23        "EB"

```

cfg.py

```

24 ]
25 conex2 = [
26     [1], [2, 3, 4, 5], [6], [7], [8], [9], [], [], [], [10], []
27 ]
28 rama3 = [
29     [0,1],
30     [0,0],
31     [0,2], [1,0], [1,1], [1,2], [2,0], [2,1], [2,2],
32     [2,0], [1,1], "EB", [2,0], [1,1], "EB", [1,1],
33     [1,0], "EB", "EB", "EB", [0,2], [1,0], [1,2], [2,0], [2,1],
34     [2,2], [1,0], "EB",
35     "EB", "EB"
36 ]
37 conex3 = [
38     [1], [2, 3, 4, 5, 6, 7, 8], [9], [10], [11], [12], [13], [14], [15], [16],
39     → [17], [], [18], [19], [], [20, 21, 22, 23, 24], [25], [], [], [], [26],
40     → [27], [27], [27], [27], [28], [29], [], [], []
41 ]
42 # Esta es la (única) rama cuando comienza la IA.
43 rama4 = [
44     [0, 0],
45     [0, 1], [0, 2], [1, 1], [1, 2], [2, 2],
46     [2, 0], [2, 0], [2, 2], [0, 2], [2, 0],
47     [1, 0], [1, 0], "EB", [0, 1], [1, 0],
48     [2, 2], [2, 2], [2, 0], [0, 2],
49     "EB", "EB", "EB", "EB"
50 ]
51 conex4 = [
52     [1, 2, 3, 4, 5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15],
53     → [16], [17], [], [18], [19], [20], [21], [22], [23], [], [], [], []
54 ]
55 # Conjunto de ramas y de conexiones.
56 ramas = [rama1, rama2, rama3, rama4]
57 conex = [conex1, conex2, conex3, conex4]
58
59 # Tableros inicializados como vacíos.
60 board = [
61     [-3, -3, -3],
62     [-3, -3, -3],
63     [-3, -3, -3]
64 ]
65 boardInt = [
66     [-3, -3, -3],
67     [-3, -3, -3],
68     [-3, -3, -3]
69 ]
70
71 # La lista sims es de simetrías y eb es estrategia básica.
72 rama = -1

```

```

74 nodo = -1
75 sims = []
76 eb = False

```

cfg.py

## Equivalencias (*equivalencias.py*)

```

1  """
2  Diferentes funciones para aplicar simetrías y para comprobar si dos tableros
3  son equivalentes.
4  """
5
6  import random
7
8
9  def simetria(boardP, sim):
10     """
11     Realiza la sim-ésima simetría al tablero.
12     Las simetrías están numeradas en sentido horario comenzando por las 12:00.
13     Casos especiales:
14     * -1: si coinciden.
15     * -2: si la simetría no existe.
16     """
17     boardC = []
18     for i in range(3):
19         boardC.append(list(boardP[i]))
20
21     if sim == 0:
22         for i in range(3):
23             boardC[i][0], boardC[i][2] = boardC[i][2], boardC[i][0]
24         return boardC
25
26     if sim == 1:
27         boardC[0][0], boardC[2][2] = boardC[2][2], boardC[0][0]
28         boardC[0][1], boardC[1][2] = boardC[1][2], boardC[0][1]
29         boardC[1][0], boardC[2][1] = boardC[2][1], boardC[1][0]
30         return boardC
31
32     if sim == 2:
33         for j in range(3):
34             boardC[0][j], boardC[2][j] = boardC[2][j], boardC[0][j]
35         return boardC
36
37     if sim == 3:
38         boardC[0][1], boardC[1][0] = boardC[1][0], boardC[0][1]
39         boardC[0][2], boardC[2][0] = boardC[2][0], boardC[0][2]
40         boardC[1][2], boardC[2][1] = boardC[2][1], boardC[1][2]
41         return boardC
42

```

```
43     if sim == -1:
44         return boardC
45
46     return -2
47
48
49 def simetriaMultiple(boardP, sims):
50     """
51     Realiza múltiples simetrías.
52     """
53     boardC = []
54     for i in range(3):
55         boardC.append(list(boardP[i]))
56
57     for sim in sims:
58         boardC = simetria(boardC, sim)
59
60     return boardC
61
62
63 def simetriaMultipleInversa(boardP, sims):
64     """
65     Realiza la inversa de una simetría múltiple.
66     """
67     boardC = []
68     for i in range(3):
69         boardC.append(list(boardP[i]))
70
71     for i in range(len(sims)):
72         boardC = simetria(boardC, sims[len(sims) - i - 1])
73
74     return boardC
75
76
77 def equivalente(boardA, boardB):
78     """
79     Comprueba si los dos tableros son equivalente y devuelve el número de la
80     simetría que convierte A en B.
81     Las simetrías están numeradas en sentido horario comenzando por las 12:00.
82     """
83     for sim in range(-1, 4):
84         boardSim = simetria(boardA, sim)
85         if boardSim == boardB:
86             return sim
87
88     return -2
89
90
91 def aleatorizar(board):
92     """
93     Añade una simetría extra (que no modifique el tablero) para hacer aleatorios
94     los movimientos.
95     """
```

```

96     posSims = [-1]
97     for sim in range(4):
98         if equivalente(board, simetria(board, sim)) == -1:
99             posSims.append(sim)
100
101     return posSims[random.randint(0, len(posSims) - 1)]

```

equivalencias.py

### Programa principal (*main.py*)

```

1  """
2  Estrategia general, importa estrategia básica.
3  """
4
5  import random, sys
6  import estrategia.cfg as cfg
7  from estrategia.equivalencias import *
8  from estrategia.basica import moveBasic, moveRandom
9
10
11 def actualiza(i, j):
12     """
13     Actualiza el tablero que ve el jugador y el resto de variables internas
14     (como el tablero que ve la máquina).
15     """
16     cfg.board[i][j] = 0
17     if cfg.eb == True:
18         return
19
20     # Si es el primer movimiento se detecta la rama inicial.
21     elif cfg.nodo == -1:
22         for i in range(3):
23             posSig = cfg.ramas[i][0]
24
25             cfg.boardInt[posSig[0]][posSig[1]] = 0
26             sim = equivalente(cfg.boardInt, cfg.board)
27             if sim != -2:
28                 cfg.sims.append(sim)
29                 cfg.rama = i
30                 cfg.nodo = 0
31                 return
32             cfg.boardInt[posSig[0]][posSig[1]] = -3
33
34     # A partir del segundo movimiento detectamos el nodo.
35     for pos in range(len(cfg.conex[cfg.rama][cfg.nodo])):
36         nodoSig = cfg.conex[cfg.rama][cfg.nodo][pos]
37         posSig = cfg.ramas[cfg.rama][nodoSig]
38
39     if (posSig == "EB"):

```

```

40         cfg.eb = True
41         return
42
43         cfg.boardInt[posSig[0]][posSig[1]] = 0
44         sim = equivalente(simetriaMultiple(cfg.board, cfg.sims), cfg.boardInt)
45
46         if sim != -2:
47             cfg.sims.append(sim)
48             cfg.nodo = nodoSig
49             return
50         cfg.boardInt[posSig[0]][posSig[1]] = -3
51
52
53 def move():
54     """
55     Decide el siguiente movimiento a realizar. Actualiza el tablero y devuelve
56     cuál es el movimiento.
57     """
58     if sys.argv[5] == "Easy":
59         if random.randint(0, 100) > 50:
60             return moveRandom(cfg.board)
61
62     if cfg.eb == True:
63         i, j = moveBasic(cfg.board)
64         return i, j
65
66     if len(cfg.conex[cfg.rama][cfg.nodo]) > 1:
67         print("Error de longitud")
68
69     cfg.nodo = cfg.conex[cfg.rama][cfg.nodo][0]
70     move = cfg.ramas[cfg.rama][cfg.nodo]
71
72     cfg.sims.append(aleatorizar(cfg.boardInt))
73
74     if move == "EB":
75         cfg.eb = True
76         i, j = moveBasic(cfg.board)
77         return i, j
78
79     # Copia del tablero para poder detectar el movimiento.
80     boardC = []
81     for i in range(3):
82         boardC.append(list(cfg.board[i]))
83
84     cfg.boardInt[move[0]][move[1]] = 1
85     cfg.board = simetriaMultipleInversa(cfg.boardInt, cfg.sims)
86
87     for i in range(3):
88         for j in range(3):
89             if boardC[i][j] != cfg.board[i][j]:
90                 return i, j
91

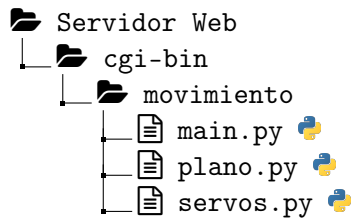
```

```
92     return -1, -1
```

```
main.py
```

## A.2. Movimiento

Aquí se recoge todo el código desarrollado en Python 🐍 relacionado con el movimiento del brazo robótico. Este código en el servidor se encuentra (de acuerdo con la figura 4 de la página 5) en el siguiente directorio:



### Programa principal (*main.py*)

```

1  """
2  Se encarga de coordinar el movimiento del brazo robótico.
3  - Define la posición espacial de las casillas del tablero y de los
4  almacenes.
5  - Contiene funciones que permiten mover piezas de una posición
6  (espacial) a otra.
7  """
8
9  from math import *
10 from movimiento.plano import verticalMove
11 from movimiento.servos import *
12
13
14 # DEFINICION DE VARIABLES
15 # Almacén más separado, si no parece que no es capaz de llegar.
16 # TODO: Revisar esto.
17 R1 = 200
18 R2 = 230
19 ang1 = 0.9*(pi/2)
20 ang2 = 0.7*(pi/2)
21
22 # V1 es un vector con la posición de 4 "X"'s
23 V1 = [[R1*cos(ang1), R1*sin(ang1)], [R1*cos(ang2), R1*sin(ang2)], [R2*cos(ang1),
24 ↪ R2*sin(ang1)], [R2*cos(ang2), R2*sin(ang2)]]
25 # U1 indica el número de pieza a coger en almacén de "X"'s
26 U1 = 0
27
28 # V2 es un vector con la posición de 4 "O"'s
29 V2 = [[R1*cos(-ang1), R1*sin(-ang1)], [R1*cos(-ang2), R1*sin(-ang2)],
30 ↪ [R2*cos(-ang1), R2*sin(-ang1)], [R2*cos(-ang2), R2*sin(-ang2)]]

```



```

29 # U1 indica el número de pieza a coger en almacén de "0"'s
30 U2 = 0
31
32 # Unión de variables del almacén.
33 V = [V1, V2]
34 U = [U1, U2]
35
36 # Posicion del tablero de casillas ancho_tablero*ancho_tablero (mm2)
37 ancho_tablero = 50
38 # Tablero también más separado.
39 # TODO: Revisar esto.
40 x_inicial_t = 80
41
42 fila_1 = [[x_inicial_t + 2*ancho_tablero, -ancho_tablero], [x_inicial_t +
↪ 2*ancho_tablero, 0], [x_inicial_t + 2*ancho_tablero, ancho_tablero]]
43 fila_2 = [[x_inicial_t + ancho_tablero, -ancho_tablero], [x_inicial_t +
↪ ancho_tablero, 0], [x_inicial_t + ancho_tablero, ancho_tablero]]
44 fila_3 = [[x_inicial_t, -ancho_tablero], [x_inicial_t, 0], [x_inicial_t,
↪ ancho_tablero]]
45
46 tablero = [fila_1, fila_2, fila_3]
47
48 # Vector con los ángulos de los servos
49 S = [0]*6
50
51
52 def reset_servos():
53     global S
54     S = [0]*6
55     moveServos(S)
56
57
58 def movePieceFromTo(p0, pf):
59     """
60     Mueve una pieza sobre el plano (horizontal) de una posición p0 = [x0, y0] a
61     una pf = [xf, yf].
62
63     La pieza utilizada es una goma Marca: Milan, Modelo: 430
64     Medidas: 2.8 x 2.8 x 1.3 cm.
65     """
66     printServosAngles(S)
67
68     # Posicionar pinza abierta por encima de la pieza (en posición de
69     # inicio).
70     r = sqrt(pow(p0[0], 2) + pow(p0[1], 2))
71     S[0] = atan(p0[1]/p0[0])
72     S[4] = -S[0] # MODIFICAR POR TEMA ANGULOS NEGATIVOS
73     ancho = 34 # Le dejo margen. Hay q vigilar q no toque a otras piezas
74     S[5] = acos((ancho+18)/52)
75     h0 = 26*sin(S[5])+68
76     phi1, phi2, phi3, phi4 = verticalMove(r, h0)
77     S[1] = phi1
78     S[2] = phi2

```

```

79
80     moveServos(S)
81     printServosAngles(S)
82
83     # Cerrar pinza para coger pieza.
84     ancho = 25 # A 25 mm. (< 28) la pinza hara fuerza - MODIFICAR
85     S[5] = acos((ancho+18)/52)
86     h0 = 26*sin(S[5])+68
87     phi1, phi2, phi3, phi4 = verticalMove(r, h0)
88     S[1] = phi1
89     S[2] = phi2
90
91     moveServos(S)
92     printServosAngles(S)
93
94     # Subir la pinza para que no se choque
95     phi1, phi2, phi3, phi4 = verticalMove(r, h0+50) # MODIFICAR
96     S[1] = phi1
97     S[2] = phi2
98     moveServos(S)
99     printServosAngles(S)
100
101     # Mover la pieza hasta la posición final
102     r = sqrt(pow(pf[0], 2) + pow(pf[1], 2))
103     S[0] = atan(pf[1]/pf[0])
104     S[4] = -S[0] # MODIFICAR POR TEMA ANGULOS NEGATIVOS
105     phi1, phi2, phi3, phi4 = verticalMove(r, h0+50)
106     S[1] = phi1
107     S[2] = phi2
108
109     moveServos(S)
110     printServosAngles(S)
111
112     # Bajar pinza sobre posición final.
113     phi1, phi2, phi3, phi4 = verticalMove(r, h0)
114     S[1] = phi1
115     S[2] = phi2
116
117     moveServos(S)
118     printServosAngles(S)
119
120     # Soltar pieza en la posición final.
121     ancho = 34 # > 28
122     S[5] = acos((ancho+18)/52)
123     h0 = 26*sin(S[5])+68
124     phi1, phi2, phi3, phi4 = verticalMove(r, h0)
125     S[1] = phi1
126     S[2] = phi2
127
128     moveServos(S)
129     printServosAngles(S)
130
131     # Subir la pinza para que no se choque

```

```

132     phi1, phi2, phi3, phi4 = verticalMove(r, h0+50) # MODIFICAR
133     S[1] = phi1
134     S[2] = phi2
135     moveServos(S)
136     printServosAngles(S)
137
138     # Dejar el brazo en posición por defecto para permitir ver el tablero.
139     # Esta posición se podría mejorar
140     reset_servos()
141     printServosAngles(S)
142
143
144 def movePiece(i, j, tipo):
145     """
146     Posiciona una pieza (de un tipo) en una posición concreta del tablero.
147     """
148     if tipo == "X":
149         tipo = 0
150     else:
151         tipo = 1
152
153     print("%.1f" % V[tipo][U[tipo]][0], end = ",")
154     print("%.1f" % V[tipo][U[tipo]][1], end = ",")
155     print("%.1f" % tablero[i][j][0], end = ",")
156     print("%.1f" % tablero[i][j][1], end = ",")
157
158     movePieceFromTo(V[tipo][U[tipo]], tablero[i][j])
159     U[tipo] += 1
160
161     print(i, end = ",")
162     print(j, end = ",")

```

main.py

## Plano (*plano.py*)

```

1  """
2  Resolución (analítica) de las ecuaciones de enlace en planos verticales.
3  - Permite pasar de posiciones en el plano a ángulos de los servomotores
4  - Se definen los parámetros del brazo.
5  """
6
7  from math import sin, cos, acos, asin, pi, sqrt
8
9
10 def resolverSistemaGeneral(p1, p2, d1, d2):
11     """
12     Resuelve el sistema:
13      $p1 = d1 \cdot \cos(b1) + d2 \cdot \cos(b2)$ 
14      $p2 = d1 \cdot \sin(b1) + d2 \cdot \sin(b2)$ 

```

```

15     Donde d1, d2, p1 y p2 son parámetros y b1, b2 son los ángulos a obtener.
16     """
17     c1 = p1*p1 + p2*p2 - d1*d1 + d2*d2
18     c2 = 2*d2*p1
19     c3 = 2*d2*p2
20     c4 = c2*c2 + c3*c3
21     c5 = 2*c1*c2
22     c6 = c1*c1 - c3*c3
23
24     if (c5*c5 - 4*c4*c6 < 0):
25         print("RAIZ COMPLEJA")
26         return []
27
28     raiz = sqrt(c5*c5 - 4*c4*c6)
29
30     aes = [(c5 + raiz)/(2*c4), (c5 - raiz)/(2*c4)]
31
32     sols = []
33     for a in aes:
34         if abs(a) <= 1:
35             b2s = [acos(a), -acos(a)]
36             for b2 in b2s:
37                 sinb1 = (p2 - d2*sin(b2))/d1
38                 if abs(sinb1) <= 1:
39                     b1s = [asin(sinb1), pi - asin(sinb1)]
40                     for b1 in b1s:
41                         sols.append([b1, b2])
42
43     return sols
44
45
46 def extraerSolucion2(phiss, phi1, phi4):
47     """
48     Devuelve una única solución que cumpla las ecuaciones del sistema 2 y con
49     los ángulos de los servos dentro del rango de funcionamiento.
50     """
51     for phis in phiss:
52         phi2 = phis[0]
53         phi3 = phis[1]
54
55         if abs(l3*cos(phi2) + l1*cos(phi3) - l2*cos(phi1) + l3*cos(phi4)) < eps
56             ↪ and \
57                 abs(l3*sin(phi2) + l1*sin(phi3) - l2*sin(phi1) + l3*sin(phi4)) < eps
58             ↪ and \
59                 phi2 >= 0 and phi2 <= pi and phi3 >= 0 and phi3 <= pi:
60                 return phi2, phi3
61
62 def resolverSistema2(phi1, phi4):
63     """
64     Resuelve el sistema:
65     
$$px = l2*cos(phi1) + l1*cos(phi4) + l4*cos(35^\circ)$$

66     
$$py = l2*sin(phi1) + l1*sin(phi4) + h$$


```

```

66     Donde l1, l2, l4 son parámetros del brazo; px, py y h son los parámetros de
67     la función y phi1, phi4 son los ángulos a obtener.
68     Solo devuelve una solución.
69     """
70     phiss = resolverSistemaGeneral(l2*cos(phi1) - l3*cos(phi4),
71                                   l2*sin(phi1) - l3*sin(phi4),
72                                   l3, l1)
73     return extraerSolucion2(phiss, phi1, phi4)
74
75
76 def extraerSolucion1(phiss, px, py):
77     """
78     Devuelve una única solución que cumpla las ecuaciones del sistema 1 y con
79     los ángulos de los servos dentro del rango de funcionamiento.
80     """
81     for phis in phiss:
82         phi1 = phis[0]
83         phi4 = phis[1]
84
85         if abs(l2*cos(phi1) + l1*cos(phi4) + l4*cos((35*pi)/180) - px) < eps and
86             ↪ \
87             abs(l2*sin(phi1) + l1*sin(phi4) + h - py) < eps and \
88             phi1 >= 0 and phi1 <= pi and phi4 <= pi/2 and phi4 >= -pi/2:
89                 return phi1, phi4
90
91 def resolverSistema1(px, py):
92     """
93     Resuelve el sistema:
94     px = l2*cos(phi1) + l1*cos(phi4) + l4*cos(35°)
95     py = l2*sin(phi1) + l1*sin(phi4) + h
96     Donde l1, l2, l4 son parámetros del brazo; px, py y h son los parámetros de
97     la función y phi1, phi4 son los ángulos a obtener.
98     Solo devuelve una solución.
99     """
100    phiss = resolverSistemaGeneral(px - l4*cos((35*pi)/180), py - h, l2, l1)
101    return extraerSolucion1(phiss, px, py)
102
103
104 def verticalMove(px, py):
105     """
106     Dada una posición (px, py) en un plano vertical, devuelve los ángulos de los
107     servos que corresponden a esa posición.
108     """
109     # Sistema 1.
110     phi1, phi4 = resolverSistema1(px, py)
111     # Sistema 2.
112     phi2, phi3 = resolverSistema2(phi1, phi4)
113     return phi1, phi2, phi3, phi4
114
115
116 # Tolerancia.
117 eps = 1e-6

```

```

118
119 # Definir parámetros del brazo robótico.
120 l1 = 160
121 l2 = 148
122 l3 = 54
123 l4 = 42
124 l5 = 68.81
125 dx = 34.4
126 dy = 24.22
127 # Consideraremos la altura como un parámetro más.
128 h = 0

```

plano.py

### Servomotores (*servos.py*)

```

servos.py
1  """
2  Define y mueve simultáneamente y de manera progresiva los servos.
3  -- Todo lo de mover los servos está por ahora comentado --
4
5  Los servos están numerados de la siguiente manera:
6  0: ROTACION
7  1: BRAZO PRINCIPAL
8  2: BRAZO SECUNDARIO
9  3: NO MUEVE NADA
10 4: PINZA ROTACIÓN
11 5: PINZA APERTURA
12 """
13
14 # from adafruit_servokit import ServoKit
15 from time import sleep
16 from math import *
17 from threading import Thread
18
19
20 # kit = ServoKit(channels = 16)
21
22 # Variable real de ángulos en servos.
23 Sp = [0]*6
24
25
26 def rad2Deg(phi):
27     """
28     Convierte de radianes a grados.
29     """
30     return (phi* 180)/pi
31
32
33 def printServosAngles(S):
34     """

```

```

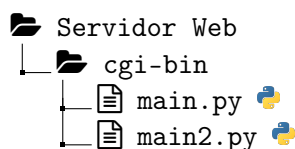
35     Devuelve información de los ángulos de los servos.
36     """
37     for i in range(6):
38         # El 3 no es un servo.
39         if i != 3:
40             print(floor(rad2Deg(S[i])), end = ",")
41
42
43 def moveServo(servo, angle):
44     """
45     Mueve el servo a un determinado ángulo (en radianes) de manera progresiva.
46     """
47     global Sp
48     angle = rad2Deg(angle)
49     steps = 50
50     time = 0 # Cambiar este valor al hacer la conexión real con el brazo.
51     timeStep = time/steps
52     angleIni = Sp[servo]
53     h = (angle - angleIni)/steps
54     for i in range(steps):
55         angleIni = angleIni + h
56         # PARA QUE NO HAYA PROBLEMAS TRUNCO!!!!!!!!
57         # kit.servo[servo].angle = floor(angleIni)
58         sleep(timeStep)
59
60     Sp[servo] = floor(angleIni)
61
62
63 def moveServos(angles):
64     """
65     Mueve los servos simultáneamente a los ángulos dados (en radianes).
66     """
67     Thread(target=moveServo, args=[0, angles[0]]).start()
68     Thread(target=moveServo, args=[1, angles[1]]).start()
69     Thread(target=moveServo, args=[2, angles[2]]).start()
70     Thread(target=moveServo, args=[4, angles[4]]).start()
71     Thread(target=moveServo, args=[5, angles[5]]).start()

```

servos.py

### A.3. Programa principal

Aquí se recoge todo el código desarrollado en Python 🐍 encargado de fusionar/coordinar el código de las secciones anteriores (A.1 y A.2). Este código en el servidor se encuentra (de acuerdo con la figura 4 de la página 5) en el siguiente directorio:



Programa principal 1 (*main1.py*)

```

1  """
2  Es el programa principal para cuando comienza a jugar el usuario, que coordina
3  la estrategia y el movimiento del brazo robótico.
4
5  Se le pasan por parámetros al ejecutarlo el estado de una partida ya comenzada,
6  entonces:
7      1. Posiciona servomotores en posición de inicio.
8      2. Mueve (físicamente) la ficha del usuario a la posición seleccionada.
9      3. Da una respuesta al tablero (de acuerdo a la dificultad).
10     4. Mueve (físicamente) la ficha de respuesta.
11     5. Devuelve diferentes datos (entre ellos las próximas url's) para que la
12     página web se pueda actualizar. Estos datos se devuelven en una línea y
13     separados por comas.
14 """
15
16 import sys
17 import estrategia.cfg as cfg
18 from estrategia.main import actualiza, move
19 from estrategia.basica import game_end
20 from estrategia.equivalencias import simetriaMultiple
21 import movimiento.main
22 from movimiento.main import movePiece, reset_servos
23
24
25 def board2Str(M):
26     """
27     Convierte el tablero a string.
28     """
29     boardStr = ""
30     for i in range(3):
31         for j in range(3):
32             if (M[i][j] == 0):
33                 boardStr += "0"
34             elif (M[i][j] == 1):
35                 boardStr += "X"
36             else:
37                 boardStr += "."
38
39     return boardStr
40
41
42 def readVariables():
43     """
44     Leer las variables y guardarlas. Devuelve la última jugada realizada.
45     """
46     movsStr = sys.argv[1]
47     movs = []
48     for i in range(int(len(movsStr)/2)):
49         movs.append([int(movsStr[2*i]), int(movsStr[2*i + 1])])
50

```



```

51     for i in range(len(movs) - 1):
52         if i%2 == 0:
53             cfg.board[movs[i][0]][movs[i][1]] = 0
54         else:
55             cfg.board[movs[i][0]][movs[i][1]] = 1
56
57     cfg.rama = int(sys.argv[2])
58     cfg.nodo = int(sys.argv[3])
59     simsStr = sys.argv[4]
60     for i in range(len(simsStr)):
61         if simsStr[i] != "-":
62             if i != 0 and simsStr[i - 1] != "-":
63                 cfg.sims.append(int(simsStr[i]))
64             elif i != 0 and simsStr[i - 1] == "-":
65                 cfg.sims.append(int(simsStr[i-1:i+1]))
66             else:
67                 cfg.sims.append(int(simsStr[i]))
68
69     cfg.boardInt = simetriaMultiple(cfg.board, cfg.sims)
70
71     if sys.argv[5] == "False":
72         cfg.eb = False
73     else:
74         cfg.eb = True
75
76     actualiza(movs[len(movs) - 1][0], movs[len(movs) - 1][1])
77
78     return movs[len(movs) - 1][0], movs[len(movs) - 1][1]
79
80
81 def nextUrl(i, j):
82     """
83     Crea e imprime la siguiente dirección web. También el tablero.
84     """
85     url = "&rama=" + str(cfg.rama)
86     url += "&nodo=" + str(cfg.nodo)
87     url += "&sims="
88     for sim in cfg.sims:
89         url += str(sim)
90     url += "&eb=" + str(cfg.eb)
91     url += "&movs=" + sys.argv[1] + str(i) + str(j)
92
93     return url
94
95
96 def printData(i, j):
97     """
98     Imprime por pantalla diferentes datos.
99     """
100     data = board2Str(cfg.board) + ","
101     data += nextUrl(i, j)
102     print(data, end = ",")
103

```

```

104
105 # Iniciar los servos.
106 reset_servos()
107 # Leer movimiento humano y mover la pieza correspondiente.
108 i, j = readVariables()
109 movePiece(i, j, "O")
110 if game_end(cfg.board) != "User wins":
111     # Decidir movimiento respuesta y mover la pieza correspondiente.
112     i, j = move()
113     # Si se puede hacer movimiento, mover la pieza.
114     if i != -1 and j != -1:
115         movePiece(i, j, "X")
116 # Devolver datos necesarios.
117 printData(i, j)
118 # Mirar si la partida ha terminado.
119 print(game_end(cfg.board))

```

main.py

## Programa principal 2 (*main2.py*)

```

main2.py
1  """
2  Es el programa principal para cuando el usuario juega segundo, que coordina la
3  estrategia y el movimiento del brazo robótico.
4
5  Se le pasan por parámetros al ejecutarlo el estado de una partida ya comenzada,
6  entonces:
7      1. Posiciona servomotores en posición de inicio.
8      2. Mueve (físicamente) la ficha del usuario a la posición seleccionada.
9      3. Da una respuesta al tablero (de acuerdo a la dificultad).
10     4. Mueve (físicamente) la ficha de respuesta.
11     5. Devuelve diferentes datos (entre ellos las próximas url's) para que la
12     página web se pueda actualizar. Estos datos se devuelven en una línea y
13     separados por comas.
14  """
15
16 import sys
17 import estrategia.cfg as cfg
18 from estrategia.main import actualiza, move
19 from estrategia.basica import game_end
20 from estrategia.equivalencias import simetriaMultiple
21 import movimiento.main
22 from movimiento.main import movePiece, reset_servos
23
24
25 def board2Str(M):
26     """
27     Convierte el tablero a string.
28     """
29     boardStr = ""

```

```

30     for i in range(3):
31         for j in range(3):
32             if (M[i][j] == 0):
33                 boardStr += "0"
34             elif (M[i][j] == 1):
35                 boardStr += "X"
36             else:
37                 boardStr += "."
38
39     return boardStr
40
41
42 def readVariables():
43     """
44     Leer las variables y guardarlas. Devuelve la última jugada realizada.
45     """
46     movsStr = sys.argv[1]
47     movs = []
48     for i in range(int(len(movsStr)/2)):
49         movs.append([int(movsStr[2*i]), int(movsStr[2*i + 1])])
50
51     for i in range(len(movs) - 1):
52         if i%2 == 0:
53             cfg.board[movs[i][0]][movs[i][1]] = 1
54         else:
55             cfg.board[movs[i][0]][movs[i][1]] = 0
56
57     cfg.rama = int(sys.argv[2])
58     cfg.nodo = int(sys.argv[3])
59     simsStr = sys.argv[4]
60     for i in range(len(simsStr)):
61         if simsStr[i] != "-":
62             if i != 0 and simsStr[i - 1] != "-":
63                 cfg.sims.append(int(simsStr[i]))
64             elif i != 0 and simsStr[i - 1] == "-":
65                 cfg.sims.append(int(simsStr[i-1:i+1]))
66             else:
67                 cfg.sims.append(int(simsStr[i]))
68
69     cfg.boardInt = simetriaMultiple(cfg.board, cfg.sims)
70
71     if sys.argv[5] == "False":
72         cfg.eb = False
73     else:
74         cfg.eb = True
75
76     if len(movs) == 1:
77         cfg.rama = -1
78
79     actualiza(movs[len(movs) - 1][0], movs[len(movs) - 1][1])
80
81     return movs[len(movs) - 1][0], movs[len(movs) - 1][1]
82

```




```

83
84 def nextUrl(i, j):
85     """
86     Crea e imprime la siguiente dirección web. También el tablero.
87     """
88     url = "&rama=" + str(cfg.rama)
89     url += "&nodo=" + str(cfg.nodo)
90     url += "&sims="
91     for sim in cfg.sims:
92         url += str(sim)
93     url += "&eb=" + str(cfg.eb)
94     url += "&movs=" + sys.argv[1] + str(i) + str(j)
95
96     return url
97
98
99 def printData(i, j):
100     """
101     Imprime por pantalla diferentes datos.
102     """
103     data = board2Str(cfg.board) + ","
104     data += nextUrl(i, j)
105     print(data, end = ",")
106
107
108 # Iniciar los servos.
109 reset_servos()
110 # Leer movimiento humano y mover la pieza correspondiente.
111 i, j = readVariables()
112 movePiece(i, j, "X")
113 if game_end(cfg.board) != "User wins":
114     # Decidir movimiento respuesta y mover la pieza correspondiente.
115     i, j = move()
116     # Si se puede hacer movimiento, mover la pieza.
117     if i != -1 and j != -1:
118         movePiece(i, j, "O")
119 # Devolver datos necesarios.
120 printData(i, j)
121 # Mirar si la partida ha terminado.
122 print(game_end(cfg.board))

```

main2.py

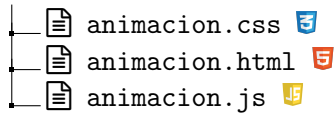
#### A.4. Animación

Aquí se recoge el código desarrollado en JavaScript  que hace posibles las animaciones del brazo robótico virtual que se muestra en la web. También se añade su correspondiente HTML  y CSS . Este código en el servidor se encuentra (de acuerdo con la figura 4 de la página 5) en el siguiente directorio:

```

└─ Servidor Web
   └─ jugar

```



### Estilos animación (*animacion.css*)

```
animacion.css
1  /* ----- */
2  /* Alzado comienza aquí. */
3  #alzado {
4      position: absolute;
5      left: 25px;
6      top: 325px;
7      transform-origin: 0 0;
8  }
9
10
11 /* Base brazo robótico. */
12 #baseA {
13     width: 50px;
14     height: 50px;
15     position: absolute;
16     bottom: 55px;
17     left: 15px;
18     background-color: #FF9900;
19 }
20
21 #tierraA {
22     width: 80px;
23     height: 65px;
24     position: absolute;
25     bottom: 0;
26     background-color: #CC0000;
27 }
28
29
30 /* Barra 1 */
31 #barra1ACont {
32     position: absolute;
33     bottom: 77.5px;
34     left: 40px;
35     transform-origin: 0 7.5px;
36 }
37
38 #barra1A {
39     width: 250px;
40     height: 15px;
41     background-color: #CC0000;
42 }
43
44
45 /* Barra 2 */
46 #barra2A {
```

```
47     width: 250px;
48     height: 15px;
49     position: absolute;
50     bottom: 77.5px;
51     left: 290px;
52     background-color: #CC0000;
53     transform-origin: 0 7.5px;
54 }
55
56
57 /* Pinza */
58 #pinzaContA {
59     position: absolute;
60     bottom: 45px;
61     left: 515px;
62 }
63
64 #pinzaA {
65     width: 10px;
66     height: 40px;
67     position: absolute;
68     top: -40px;
69     left: 20px;
70     background-color: #CC0000;
71 }
72
73 #piezaPinzaA {
74     width: 30px;
75     height: 20px;
76     position: absolute;
77     left: 10px;
78     top: 12px;
79     background-color: #2B6587;
80     display: none;
81 }
82
83 #tenaza {
84     width: 50px;
85     height: 10px;
86     position: absolute;
87     top: 390px;
88     left: 530px;
89     background-color: #0000FF;
90 }
91
92 #tenazas_hA {
93     width: 50px;
94     height: 10px;
95     position: absolute;
96     top: 0;
97     left: 0;
98     background-color: #0000FF;
99 }
```

```
100
101 #tenazas_v1A {
102     width: 10px;
103     height: 40px;
104     position: absolute;
105     top: 0;
106     left: 0;
107     background-color: #0000FF;
108     transform-origin: 0 0;
109 }
110
111 #tenazas_v2A {
112     width: 10px;
113     height: 40px;
114     position: absolute;
115     top: 0;
116     left: 40px;
117     background-color: #0000FF;
118 }
119
120
121 /* Articulaciones */
122 #articulacion {
123     width: 30px;
124     height: 30px;
125     border-radius: 50%;
126     background-color: Black;
127 }
128
129 #articulacionPos1A {
130     position: absolute;
131     left: 235px;
132     bottom: -7.5px;
133 }
134
135 #articulacionPos2A {
136     position: absolute;
137     left: 10px;
138     bottom: 25px;
139 }
140
141
142 /* Almacén de piezas. */
143 #almacenA * {
144     width: 30px;
145     height: 20px;
146     position: absolute;
147     left: 406px;
148     background-color: #2b6587;
149 }
150
151 #almacenA {
152     opacity: 0.1;
```

```
153     transition: opacity 2s;
154     transition-timing-function: easy-in;
155 }
156
157 #almacenPieza0A {
158     bottom: 72px;
159 }
160
161 #almacenPieza1A {
162     bottom: 48px;
163 }
164
165 #almacenPieza2A {
166     bottom: 24px;
167 }
168
169 #almacenPieza3A {
170     bottom: 0;
171 }
172
173
174 /* Piezas tablero. */
175 #tableroA {
176     opacity: 0.1;
177     transition: opacity 2s;
178     transition-timing-function: easy-in;
179 }
180
181 #tableroA * {
182     width: 30px;
183     height: 20px;
184     position: absolute;
185     bottom: 0;
186     background-color: #2B6587;
187     display: none;
188 }
189
190 #tableroPieza0A {
191     left: 170px;
192 }
193
194 #tableroPieza1A {
195     left: 215px;
196 }
197
198 #tableroPieza2A {
199     left: 260px;
200 }
201
202 /* Alzado termina aquí.*/
203 /* ----- */
204
205
```



```
206  /* ----- */
207  /* Planta comienza aquí. */
208  #planta {
209      position: absolute;
210      left: 500px;
211      top: 325px;
212      transform-origin: 0 0;
213  }
214
215  #robotP {
216      position: absolute;
217      bottom: 0px;
218      left: 0px;
219      transform-origin: 40px -35px;
220      z-index: 20;
221  }
222
223
224  /* Base brazo robótico. */
225  #baseP {
226      width: 50px;
227      height: 50px;
228      position: absolute;
229      bottom: 10px;
230      left: 15px;
231      background-color: #FF9900;
232      z-index: 10;
233  }
234
235  #tierraP {
236      width: 80px;
237      height: 70px;
238      position: absolute;
239      bottom: 0;
240      background-color: #CC0000;
241      z-index: 1;
242  }
243
244
245  /* Barra 1. */
246  #barra1ContP {
247      position: absolute;
248      bottom: 27.5px;
249      left: 40px;
250      z-index: 12;
251  }
252
253  #barra1P {
254      width: 250px;
255      height: 15px;
256      background-color: #CC0000;
257  }
258
```

```
259
260 /* Barra 2. */
261 #barra2ContP {
262     position: absolute;
263     bottom: 27.5px;
264     left: 290px;
265 }
266
267 #barra2P {
268     width: 250px;
269     height: 15px;
270     background-color: #CC0000;
271     left: 0;
272     bottom: 0;
273 }
274
275
276 /* Pinza. */
277 #pinzaContP {
278     position: absolute;
279     bottom: 35px;
280     left: 540px;
281     transform-origin: 0 0;
282     z-index: -10;
283 }
284
285 #pinzaP {
286     width: 40px;
287     height: 24px;
288     position: absolute;
289     bottom: -12px;
290     left: -20px;
291     background-color: #0000FF;
292 }
293
294 #piezaPinzaP {
295     width: 60px;
296     height: 60px;
297     position: absolute;
298     left: -30px;
299     top: -30px;
300     background-color: #2B6587;
301     z-index: -10;
302     /* display: none; */
303 }
304
305
306 /* Articulaciones. */
307 #articulacionPos1P {
308     position: absolute;
309     left: 235px;
310     bottom: -7.5px;
311 }
```

```
312
313 #articulacionPos2P {
314     position: absolute;
315     left: 235px;
316     bottom: -7.5px;
317 }
318
319 #articulacionPos3P {
320     position: absolute;
321     left: 25px;
322     bottom: 20px;
323     z-index: 15;
324 }
325
326
327 /* Almacenes de piezas. */
328 #almacenXsP {
329     position: absolute;
330     top: 65px;
331     left: 300px;
332     height: 30px;
333     width: 30px;
334     background-color: #336600;
335     border-radius: 50%;
336 }
337
338 #almacenOsP {
339     position: absolute;
340     top: -130px;
341     left: 300px;
342     height: 30px;
343     width: 30px;
344     background-color: #3862E0;
345     border-radius: 50%;
346 }
347
348 /* Planta termina aquí.*/
349 /* ----- */
350
351
352 /* Control de velocidad. */
353 #velocidad {
354     position: relative;
355     top: 420px;
356 }
357
358
359 /* Cerrar pantalla completa. */
360 #cerrarPant {
361     position: absolute;
362     top: 15px;
363     right: 20px;
364     font-size: 60px;
```

```

365     text-decoration: none;
366     color: #818181;
367     transition: .3s;
368 }
369
370 #cerrarPant:hover {
371     color: Black;
372 }
373
374
375 /* Mostrar pantalla completa. */
376 #pantCompl {
377     position: absolute;
378     top: 275px;
379     left: 150px;
380 }

```

animacion.css

### Animación HTML (*animacion.html*)

animacion.html

```

1  <div id="alzado">
2      <div id="barra2A"></div>
3
4      <div id="barra1ACont">
5          <div id="barra1A"></div>
6          <div id="articulacionPos1A">
7              <div id="articulacion"></div>
8          </div>
9      </div>
10
11     <div id="baseA"></div>
12     <div id="tierraA"></div>
13
14     <div id="pinzaContA">
15         <div id="pinzaA"></div>
16         <div id="tenazas_hA"></div>
17         <div id="tenazas_v1A"></div>
18         <div id="tenazas_v2A"></div>
19         <div id="articulacionPos2A">
20             <div id="articulacion"></div>
21         </div>
22         <div id="piezaPinzaA"></div>
23     </div>
24
25     <div id="almacenA">
26         <div id="almacenPieza0A"></div>
27         <div id="almacenPieza1A"></div>
28         <div id="almacenPieza2A"></div>
29         <div id="almacenPieza3A"></div>

```

```

30     </div>
31
32     <div id="tableroA">
33         <div id="tableroPieza0A"></div>
34         <div id="tableroPieza1A"></div>
35         <div id="tableroPieza2A"></div>
36     </div>
37 </div>
38
39
40 <div id="planta">
41     <div id="almacenXsP"></div>
42     <div id="almacenOsP"></div>
43
44     <div id="robotP">
45         <div id="barra2ContP">
46             <div id="barra2P"></div>
47             <div id="articulacionPos2P">
48                 <div id="articulacion"></div>
49             </div>
50         </div>
51
52         <div id="barra1ContP">
53             <div id="barra1P"></div>
54             <div id="articulacionPos1P">
55                 <div id="articulacion"></div>
56             </div>
57         </div>
58
59         <div id="articulacionPos3P">
60             <div id="articulacion"></div>
61         </div>
62
63         <div id="pinzaContP">
64             <div id="pinzaP"></div>
65             <div id="piezaPinzaP"></div>
66         </div>
67     </div>
68
69     <div id="baseP"></div>
70     <div id="tierraP"></div>
71 </div>

```

animacion.html

### Animación JS (*animacion.js*)

```

1 // Funciones para web.
2 function mostrarAnimacion() {
3     botonControl.style.border = "";

```

animacion.js

```
4     animacion.style.visibility = "visible";
5     planta.style.visibility = "hidden";
6     alzado.style.top = "250px";
7     alzado.style.transform = "scale(.7, .7)";
8     animacion.style.height = "300px";
9     animacion.style.width = "500px";
10    animacion.style.top = "120px";
11    animacion.style.left = "50px";
12    animacion.style.background = "#F0F0F0";
13    animacion.style.border = "";
14    velocidad.style.display = "none";
15    botonAnimacion.style.border = "3px inset Black";
16    cerrarPant.style.display = "none";
17    pantCompl.style.display = "block";
18    comenzarAnimacion();
19 }
20
21
22 function mostrarAnimacionCompleta() {
23     planta.style.visibility = "visible";
24     alzado.style.top = "325px";
25     alzado.style.transform = "scale(.8, .8)";
26     animacion.style.height = "530px";
27     animacion.style.width = "96.5%";
28     animacion.style.top = "10px";
29     animacion.style.left = "17px";
30     animacion.style.border = "3px solid Black";
31     animacion.style.background = "##E6E6E6";
32     velocidad.style.display = "block";
33     cerrarPant.style.display = "block";
34     pantCompl.style.display = "none";
35 }
36
37
38 function cerrarAnimacion() {
39     control.style.visibility = "visible";
40     animacion.style.visibility = "hidden";
41     planta.style.visibility = "hidden";
42     mostrarControles();
43 }
44
45
46 function mostrarControles() {
47     animacion.style.visibility = "hidden";
48     control.style.visibility = "visible";
49     botonControl.style.border = "3px inset Black";
50     botonAnimacion.style.border = "";
51 }
52
53
54 // Funciones para movimiento del brazo robótico virtual.
55 function sleep(delay) {
56     return new Promise(resolve => setTimeout(resolve, delay));
```

```

57 }
58
59
60 async function rotacionP(phi0, phif) {
61     sleepTime = 1.35*delay*Math.abs(phi0 - phif);
62
63     var phi = phi0;
64     var inc = 1;
65
66     if (phi0 > phif)
67         inc = -1;
68
69     while (phi != phif) {
70         await sleep(delay);
71         phi += inc;
72         robotP.style.transform = "rotate(" + (-phi) + "deg)";
73         pinzaContP.style.transform = "rotate(" + phi + "deg)";
74     }
75 }
76
77
78 async function move_barra1(phi0, phif) {
79     var phi = phi0;
80     var inc = 1;
81     if (phi0 > phif)
82         inc = -1;
83
84     while (phi != phif) {
85         await sleep(delay);
86         phi += inc;
87
88         barra1ACont.style.transform = "rotate(-" + phi + "deg)";
89
90         var xBarra2A = xBarra2AIniA + length1A*(Math.cos(phi*Math.PI/180) - 1);
91         var yBarra2A = yBarra2AIniA + length1A*Math.sin(phi*Math.PI/180);
92         barra2A.style.left = xBarra2A + "px";
93         barra2A.style.bottom = yBarra2A + "px";
94
95         var xPinza = posPinzaA[0] + length1A*(Math.cos(phi*Math.PI/180) -
96             ↪ Math.cos(phi0*Math.PI/180));
97         var yPinza = posPinzaA[1] + length1A*(Math.sin(phi*Math.PI/180) -
98             ↪ Math.sin(phi0*Math.PI/180));
99         pinzaContA.style.left = xPinza + "px";
100        pinzaContA.style.bottom = yPinza + "px";
101
102        var length1P = length1A*(Math.cos(phi*Math.PI/180));
103        barra1P.style.width = length1P + "px";
104        articulacionPos1P.style.left = length1P - 15 + "px";
105        barra2ContP.style.left = length1P + 40 + "px";
106        pinzaContP.style.left = 40 + length1P +
107            ↪ document.getElementById("barra2P").offsetWidth + "px";
108    }
109 }

```

```

107     posPinzaA = [xPinza, yPinza];
108 }
109
110
111 async function move_barra2(phi0, phif) {
112     var phi = phi0;
113     var inc = 1;
114     if (phi0 > phif)
115         inc = -1;
116
117     var transPinzaIni = document.getElementById("pinzaContA").style.transform;
118     while (phi != phif) {
119         await sleep(delay);
120         phi += inc;
121
122         barra2A.style.transform = "rotate(" + phi + "deg)";
123
124         var xPinza = posPinzaA[0] + length2A*(Math.cos(phi*Math.PI/180) -
125             ↪ Math.cos(phi0*Math.PI/180));
126         var yPinza = posPinzaA[1] + length2A*(-Math.sin(phi*Math.PI/180) +
127             ↪ Math.sin(phi0*Math.PI/180));
128         pinzaContA.style.left = xPinza + "px";
129         pinzaContA.style.bottom = yPinza + "px";
130
131         var length2P = length2A*(Math.cos(phi*Math.PI/180));
132         barra2P.style.width = length2P + "px";
133         articulacionPos2P.style.left = length2P - 15 + "px";
134         pinzaContP.style.left = 40 +
135             ↪ document.getElementById("barra1P").offsetWidth + length2P + "px";
136     }
137
138     posPinzaA = [xPinza, yPinza];
139 }
140
141
142 async function move_barras(phi1, phi2) {
143     var sleepTime1 = 2*delay*Math.abs(phi1 - angulosBarrasA[0]);
144     var sleepTime2 = 2*delay*Math.abs(phi2 - angulosBarrasA[1]);
145     sleepTime = 1.1*(sleepTime1 + sleepTime2);
146
147     move_barra1(angulosBarrasA[0], phi1);
148     await sleep(sleepTime1);
149     move_barra2(angulosBarrasA[1], phi2);
150     await sleep(sleepTime2);
151     angulosBarrasA = [phi1, phi2];
152 }
153
154
155 async function move_piece(noAlmacen, noTablero) {
156     if (noAlmacen >= 0)
157         almacenAngP = Math.abs(almacenAngP);
158     else {
159         almacenAngP = -Math.abs(almacenAngP);
160     }
161 }

```



```

157     noAlmacen *= -1;
158 }
159
160 // Ir al almacén.
161 rotacionP(0, almacenAngP);
162 await sleep(sleepTime);
163
164 tableroA.style.opacity = "0.1";
165 move_barras(angulosBarrasA[0] - 1, 20);
166 await sleep(sleepTime);
167
168 almacenA.style.opacity = "1";
169 move_barras(almacenAngsA[noAlmacen][0], almacenAngsA[noAlmacen][1]);
170 await sleep(sleepTime);
171 document.getElementById("piezaPinzaA").style.display = "block";
172 document.getElementById("almacenPieza" + noAlmacen + "A").style.display =
    ↪ "none";
173
174 // Ir al tablero.
175 rotacionP(almacenAngP, 0);
176 await sleep(sleepTime);
177
178 almacenA.style.opacity = "0.1";
179 tableroA.style.opacity = "1";
180 move_barras(tableroAngsA[noTablero][0], tableroAngsA[noTablero][1]);
181 await sleep(sleepTime);
182 piezaPinzaA.style.display = "none";
183 document.getElementById("tableroPieza" + noTablero + "A").style.display =
    ↪ "block";
184 }
185
186
187 async function reset() {
188     move_barras(90, 73);
189     await sleep(sleepTime);
190 }
191
192
193 // Función para probar.
194 async function comenzarAnimacion() {
195     await sleep(500);
196     reset();
197     await sleep(280*delay + 500);
198
199     move_piece(0, 0);
200     await sleep(450*delay);
201
202     move_piece(-1, 2);
203     await sleep(450*delay);
204
205     move_piece(2, 1);
206     await sleep(450*delay);
207

```





```

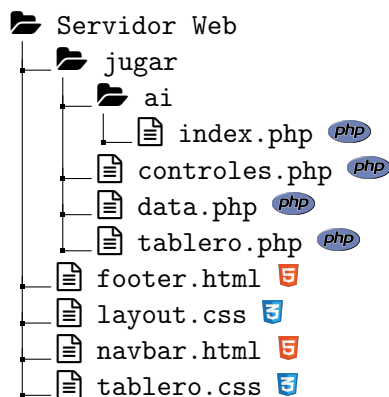
208     reset();
209 }
210
211
212 var length1A = document.getElementById("barra1A").offsetWidth;
213 var length2A = document.getElementById("barra2A").offsetWidth;
214
215 var xBarra2AIniA =
  ↳ getComputedStyle(document.getElementById("barra2A")).getPropertyValue("left");
216 var yBarra2AIniA =
  ↳ getComputedStyle(document.getElementById("barra2A")).getPropertyValue("bottom");
217 // Conversión.
218 xBarra2AIniA = Number(xBarra2AIniA.slice(0, xBarra2AIniA.length - 2))
219 yBarra2AIniA = Number(yBarra2AIniA.slice(0, yBarra2AIniA.length - 2))
220
221 var delay = 25 - Number(document.getElementById("velSlider").value);
222 var sleepTime = 0;
223
224 var posPinzaA = [515, 45];
225 var angulosBarrasA = [0, 0];
226
227 var almacenAngsA = [[48, 31], [45, 35], [42, 39], [39, 42]];
228 var tableroAngsA = [[71, 75], [67, 68], [61, 63]];
229
230 var almacenAngP = 45;

```

animacion.js

## A.5. Servidor

El resto de la parte de archivos del servidor han sido desarrollados en diversos lenguajes, como son PHP , HTML , CSS  y JavaScript . En este caso, debido a la extensión, solo se mostrarán los archivos considerados más representativos. Más concretamente se mostrarán los recogidos en el siguiente árbol (de acuerdo con la figura 4 de la página 5).



El contenido de los archivos a continuación se muestra siguiendo el mismo orden en el que aparecen en el árbol anterior.

Página principal jugar/ai/ (*index.php*)

```

index.php
1  <!DOCTYPE HTML>
2
3  <html>
4      <head>
5          <meta charset="UTF-8">
6          <title>Tres en Raya</title>
7          <link rel="stylesheet" type="text/css" href="../../layout.css" />
8          <link rel="stylesheet" type="text/css" href="../../tablero.css" />
9          <link rel="stylesheet" type="text/css" href="../../animacion.css" />
10         <style type="text/css">
11             .data {
12                 width: 90%;
13                 min-width: 500px;
14                 height: 70px;
15                 text-align: center;
16             }
17         </style>
18     </head>
19
20
21     <body>
22         <header id="header">
23             <div style="float: left;">
24                 <a href="/proyecto/" title="Página principal.">
25                     <h1>Tres en Raya</h1>
26                 </a>
27             </div>
28             <div style="float: right;">
29                 <a href="/proyecto/jugar/ai/" title="Tú juegas segundo.">
30                     <h3>Empieza el brazo</h3>
31                 </a>
32             </div>
33             <div style="margin: 0 auto; width: 100px;" title="Juega contra la
↪ máquina.">
34                 <a href="/proyecto/jugar/">
35                     <h1>Jugar</h1>
36                 </a>
37             </div>
38         </header>
39
40
41         <?php
42         include("../../navbar.html")
43         ?>
44
45
46         <main id="main">
47             <div class="innertube">
48                 <p>Juega al tres en raya, tú vas segundo (también puedes
49                 <a href="/proyecto/jugar/per/" title="Empieza tú a jugar.">

```

```

50         empezar tú</a>).
51     </p>
52 </div>
53
54 <?php
55 $dif = 'alta';
56 if (isset($_GET['dif']))
57     $dif = $_GET['dif'];
58
59 if (isset($_GET['movs'])) {
60     $movs = $_GET['movs'];
61     $rama = $_GET['rama'];
62     $nodo = $_GET['nodo'];
63     $sims = $_GET['sims'];
64     $eb = $_GET['eb'];
65
66     if ($dif == 'baja')
67         $eb = 'Easy';
68     else if ($dif == 'media')
69         $eb = 'True';
70
71     $command = 'python3 ../../cgi-bin/main2.py '.$movs.' '.$rama.'
72     ↪ '.$nodo.' '.$sims.' '.$eb;
73     $output = exec($command);
74     $outputArray = split(" ", $output);
75     $n = sizeof($outputArray);
76
77     $board = $outputArray[$n - 3];
78     $url = '?dif='.$dif.$outputArray[$n - 2];
79 }
80 else {
81     $rand = rand(1, 4);
82     if ($rand == 1) {
83         $movs = "00";
84         $sims = -1;
85         $board = 'X.....';
86     }
87     else if ($rand == 2) {
88         $movs = "02";
89         $sims = 0;
90         $board = '..X.....';
91     }
92     else if ($rand == 3) {
93         $movs = "20";
94         $sims = 2;
95         $board = '.....X..';
96     }
97     else {
98         $movs = "22";
99         $sims = 1;
100         $board = '.....X';
101     }
102     $url =
103     ↪ '?dif='.$dif.'&rama=3&nodo=0&sims='.$sims.'&eb=False&movs='.$movs;

```

```

102         $outputArray = array('Not ended');
103         $n = sizeof($outputArray);
104     }
105     ?>
106
107     <?php
108     include("../tablero.php");
109     $page = 'ai';
110     include("../controles.php");
111     include("../data.php");
112     ?>
113
114     <div id="animacion">
115         <?php
116         include("../animacion.html");
117         ?>
118
119         <div id="velocidad" align="center">
120             <p>Ajusta la velocidad:</p>
121             <form action="/proyecto/jugar/ai/">
122                 <?php
123                 echo '<input type="range" id="velSlider" name="vel" min="5"
124                 ↪ max="20" ' ;
125                 if (isset($_GET['vel']))
126                     echo ' value="'. $_GET['vel']. ' ' ;
127                 echo 'step="3" onchange="delay = 25 - this.value;" />';
128                 ?>
129                 <input type="submit" value="Vuelve a empezar" />
130             </form>
131         </div>
132
133         <button id="pantCompl" onclick="mostrarAnimacionCompleta();">
134             Pantalla completa
135         </button>
136         <a id="cerrarPant" href="javascript:void(0)"
137         ↪ onclick="cerrarAnimacion();">
138             &times;
139         </a>
140     </div>
141 </main>
142
143     <?php
144     include("../../footer.html")
145     ?>
146
147     <script src="../animacion.js"></script>
148     <script type="text/javascript">
149         <?php
150         if (isset($_GET['vel'])) {
151             echo 'mostrarAnimacion();';
152             echo 'mostrarAnimacionCompleta();';
153         }
154     </script>
155     ?>
156
157     </body>
158 </html>

```

```

153         else
154             echo 'mostrarControles()';
155     ?>
156     </script>
157
158     </body>
159 </html>

```

index.php

### Controles (*controles.php*)

```

1 <style type="text/css">
2     #endMessage {
3         font-weight: bold;
4         font-size: 30px;
5     }
6     input[type = radio] {
7         margin-left: 20px;
8     }
9 </style>
10
11
12 <div class="control" align="center" id="control">
13     <button id="botonControl" onclick="mostrarControles();">
14         Controles
15     </button>
16     <button id="botonAnimacion" onclick="mostrarAnimacion();">
17         Animación
18     </button>
19
20     <?php
21     if ($outputArray[$n - 1] == "AI wins")
22         echo '<p id="endMessage" style="color: red;">Lo sentimos, has
23             ↳ perdido.</p>';
24     else if ($outputArray[$n - 1] == "Tie")
25         echo '<p id="endMessage">Ha sido un empate.</p>';
26     else if ($outputArray[$n - 1] == "User wins")
27         echo '<p id="endMessage" style="color: green;">Enhorabuena, has
28             ↳ ganado.</p>';
29     else
30         echo '<br /><br />';
31     ?>
32
33     <p>Vuelve a empezar, vacía el tablero.</p>
34     <?php
35     echo '<a href="/proyecto/jugar/' . $page . '/' . $dif . '>';
36     echo $dif . '>';
37     echo '<button>Vacía</button></a>';
38     ?>

```

```

37
38 <p>Ajusta la dificultad.</p>
39 <div align="center">
40     <?php
41     echo '<form action="/proyecto/jugar/' . $page . '/' . ">';
42     $same = '<input type="radio" name="dif" value=';
43     $dif = 'alta';
44     if (isset($_GET['dif']))
45         $dif = $_GET['dif'];
46     $values = array('baja', 'media', 'alta');
47     $valuesDisp = array('Fácil', 'Medio', 'Imposible');
48     for ($i = 0; $i < 3; ++$i) {
49         if ($values[$i] == $dif)
50             echo '<input checked type="radio" name="dif"
51                 ↪ value="' . $values[$i] . '" />' . $valuesDisp[$i];
52             else
53                 echo '<input type="radio" name="dif" value="' . $values[$i] . '"
54                     ↪ />' . $valuesDisp[$i];
55     }
56     echo '<br /><br />';
57     echo '<input type="submit" value="Actualiza cambios" />';
58     echo '</form>';
59     ?>
60 </div>
61 </div>
62
63 <?php
64 if ($outputArray[$n - 1] == "User wins")
65     echo '
66     <canvas id="canvas"></canvas>
67     <script type="text/javascript" src="../confeti.js"></script>
68     <style type="text/css">
69         canvas {
70             position: absolute;
71             top: 0;
72             left: 0;
73             display: block;
74             z-index: -1;
75         }
76     </style>';
77     ?>

```

controles.php

### Tabla datos diversos (*data.php*)

```

1 <div align="center">
2     <br />
3     <h2>Ángulos servos última jugada</h2>

```

data.php

```

4      <br />
5      <table border="1" class="data">
6          <tr>
7              <th></th>
8              <th>ROTACIÓN (0)</th>
9              <th>BRAZO PRINCIPAL (1)</th>
10             <th>BRAZO SECUNDARIO (2)</th>
11             <th>PINZA ROTACIÓN (4)</th>
12             <th>PINZA APERTURA (5)</th>
13         </tr>
14         <tr>
15             <td colspan="6">Movimiento humano.</td>
16         </tr>
17
18         <?php
19         for ($i = 0; $i < 9; ++$i) {
20             echo '<tr>';
21             echo '<td><b>Mov ' . $i . '</b></td>';
22             for ($j = 0; $j < 5; ++$j)
23                 echo '<td>' . $outputArray[5*$i + $j + 4] . '</td>';
24             echo '</tr>';
25         }
26         ?>
27         <tr>
28             <td colspan="6">Movimiento brazo.</td>
29         </tr>
30         <?php
31         for ($ip = 9; $ip < 18; ++$ip) {
32             echo '<tr>';
33             $ipp = $ip - 9;
34             echo '<td><b>Mov ' . $ipp . '</b></td>';
35             for ($jp = 0; $jp < 5; ++$jp)
36                 echo '<td>' . $outputArray[5*$ip + $jp + 10] . '</td>';
37             echo '</tr>';
38         }
39         ?>
40     </table>
41
42     <br />
43     <ul align="center">
44         <li><b>Mov 0:</b> Posición predeterminada.</li>
45         <li><b>Mov 1:</b> Pinza sobre la pieza a coger en el almacén (a la altura
↪   justa para que al cerrar la pinza coja la pieza).</li>
46         <li><b>Mov 2:</b> Justo después de cerrar la pinza.</li>
47         <li><b>Mov 3:</b> Subir la pinza para que no se choque.</li>
48         <li><b>Mov 4:</b> Sobre la posición final.</li>
49         <li><b>Mov 5:</b> Pinza bajada sobre posición final.</li>
50         <li><b>Mov 6:</b> Pieza soltada en la posición final (con las pinzas
↪   abiertas).</li>
51         <li><b>Mov 7:</b> Subir la pinza para que no se choque.</li>
52         <li><b>Mov 8:</b> Posición predeterminada.</li>
53     </ul>
54

```



```

55
56 <br /><br /><br /><br />
57 <h2>Movimiento piezas última jugada</h2>
58 <br />
59 <table border="1" class="data">
60   <tr>
61     <th>PIEZA COGIDA DE (ejes)</th>
62     <th>PIEZA COGIDA DE (número)*</th>
63     <th>PIEZA DEJADA EN (ejes)</th>
64     <th>PIEZA DEJADA EN (tablero)**</th>
65   </tr>
66   <tr>
67     <td colspan="4">Movimiento humano (mueve O's).</td>
68   </tr>
69   <tr>
70     <td>x = <?php echo $outputArray[0]; ?> , y = <?php echo $outputArray[1]
↪ <?> [mm]</td>
71     <td>Falta, pero va por orden.</td>
72     <td>x = <?php echo $outputArray[2]; ?> , y = <?php echo $outputArray[3]
↪ <?> [mm]</td>
73     <td>Fila = <?php echo $outputArray[49] + 1; ?> - Columna = <?php echo
↪ $outputArray[50] + 1; ?></td>
74   </tr>
75   <tr>
76     <td colspan="4">Movimiento brazo (mueve X's).</td>
77   </tr>
78   <tr>
79     <td>x = <?php echo $outputArray[51]; ?> , y = <?php echo
↪ $outputArray[52] ?> [mm]</td>
80     <td>Falta, pero va por orden.</td>
81     <td>x = <?php echo $outputArray[53]; ?> , y = <?php echo
↪ $outputArray[54] ?> [mm]</td>
82     <td>Fila = <?php echo $outputArray[100] + 1; ?> - Columna = <?php echo
↪ $outputArray[101] + 1; ?></td>
83   </tr>
84 </table>
85
86 <br />
87 <ul>
88   <li><b>*</b>Número de pieza en el almacén correspondiente (hay 4 huecos en
↪ cada almacén).</li>
89   <li><b>*</b>Posición relativa en el tablero (fila y columna).</li>
90   <li>Los ejes parten del brazo (que está puesto donde está el título de
91     la página, es decir, detrás del tablero). Eje de abscisas horizontal (en
92     pantalla de ordenador) y de ordenadas vertical (ídem).</li>
93 </ul>
94
95 <br /><br /><br />
96 </div>

```

data.php

Tablero del Tres en Raya (*tablero.php*)

```

1  <table id="board">
2      <?php
3      for ($i = 0; $i < 3; ++$i) {
4          echo '<tr>';
5          for ($j = 0; $j < 3; ++$j) {
6              if ($board[3*$i + $j] != ".")
7                  echo '<td id="' . $board[3*$i + $j] . '">' . $board[3*$i + $j] . '</td>';
8              else if ($outputArray[$n - 1] != "Not ended")
9                  echo '<td></td>';
10             else
11                 echo '<td><a href="' . $url.$i.$j . '"><button
12                     id="tic"><span></span></button></a></td>';
13             }
14         echo '</tr>';
15     }
16 </table>

```

Pie de página (*footer.html*)

```

1  <footer id="footer">
2      <p align="center" style="font-variant: small-caps; font-size: 1.1em;">
3          <span style="float: left;">Proyecto II</span>
4          David Álvarez - Guillermo Creus
5          <span style="float: right;">
6              Barcelona, mayo de 2019
7          </span>
8      </p>
9  </footer>

```

Estilos principal (*layout.css*)

```

1  body {
2      margin: 0;
3      padding: 0;
4      font-family: Sans-serif;
5      line-height: 1.5em;
6  }
7
8
9  /* Cabecera. */

```

```
10 #header {
11     width: 100%;
12     height: 75px;
13     background: #3366FF;
14 }
15
16 #header h1 {
17     margin: 0;
18     padding: 25px 15px 0 15px;
19 }
20
21 #header h3 {
22     margin: 0;
23     padding: 28px 20px 0 20px;
24 }
25
26 #header a {
27     text-decoration: none;
28     color: Black;
29 }
30
31
32 /* Barra de navegación. */
33 #nav {
34     position: fixed;
35     top: 75px;
36     bottom: 0;
37     width: 230px;
38     background: #99B3FF;
39 }
40
41 #nav ul {
42     margin-top: -10px;
43     padding-bottom: 10px;
44 }
45
46 #nav h1 a {
47     text-decoration: none;
48     color: Black;
49 }
50
51 main {
52     position: fixed;
53     top: 75px;
54     bottom: 35px;
55     left: 230px;
56     right: 0;
57     min-width: 600px;
58     background: #F0F0F0;
59     overflow: auto;
60 }
61
62 .control {
```

```
63     margin: 35px 0 0 100px;
64     height: 360px;
65     width: 300px;
66 }
67
68 img {
69     padding: 15px;
70     width: 70px;
71 }
72
73 img:hover {
74     transform: scale(1.1);
75     transition: transform .2s linear .1s;
76 }
77
78
79 /* Pie de página. */
80 #footer {
81     position: fixed;
82     bottom: 0px;
83     left: 230px;
84     right: 0px;
85     height: 35px;
86     background: #ccc;
87 }
88
89 #footer p {
90     margin: 4px 30px 0 30px;
91 }
92
93 #footer a {
94     text-decoration: none;
95     color: Black;
96 }
97
98
99 /* Animación. */
100 #animacion {
101     position: absolute;
102     background-color: #E6E6E6;
103     transition: all 1s;
104 }
105
106
107 /* Globales. */
108 .innertube {
109     margin: 20px;
110 }
111
112 html {
113     overflow: hidden;
114 }
115
```

```
116 button, input {
117     cursor: pointer;
118 }
119
120
121 /* Actualizar con el tamaño. */
122 @media only all
123 and (max-width: 900px) {
124     .control {
125         width: 95%;
126         height: auto;
127         margin: 20px 0 0 25px;
128         padding-bottom: 50px;
129     }
130     .end {
131         width: 95%;
132         height: auto;
133         margin: 20px 0 0 25px;
134         padding-bottom: 50px;
135     }
136 }
137
138 @media only all
139 and (max-width: 1050px)
140 and (min-width: 899px) {
141     .control {
142         margin: 60px 0 0 15px;
143     }
144     .end {
145         margin: 30px 0 0 15px;
146     }
147 }
148
149 @media only all
150 and (max-width: 1200px)
151 and (min-width: 1049px){
152     .control {
153         margin: 60px 0 0 70px;
154     }
155     .end {
156         margin: 30px 0 0 70px;
157     }
158 }
```

layout.css

### Barra de navegación (*navbar.html*)

```
1 <nav id="nav">
2   <div class="innertube">
```

navbar.html

```
3
4     <h1>
5         <a href="/proyecto/proyecto/" title="El proyecto.">
6             Proyecto
7         </a>
8     </h1>
9     <ul>
10         <li>
11             <a href="/proyecto/proyecto/documentacion/" title="La documentación
12                 ↳ del proyecto.">
13                 Documentación
14             </a>
15         </li>
16         <li>
17             <a href="/proyecto/proyecto/codigo/" title="Todo el código.">
18                 El código
19             </a>
20         </li>
21     </ul>
22
23     <h1>
24         <a href="/proyecto/jugar/" title="Juega contra la máquina.">
25             Jugar
26         </a>
27     </h1>
28     <ul>
29         <li>
30             <a href="/proyecto/jugar/per/" title="Empieza tú a jugar.">
31                 Empieza tú
32             </a>
33         </li>
34         <li>
35             <a href="/proyecto/jugar/ai/" title="Tú juegas segundo.">
36                 Empieza el brazo
37             </a>
38         </li>
39         <li>
40             <a href="/proyecto/jugar/ver/" title="Ve una partida.">
41                 Ver partida
42             </a>
43         </li>
44     </ul>
45
46     <h1>
47         <a href="/proyecto/presentaciones/" title="Todas las presentaciones.">
48             Presentaciones
49         </a>
50     </h1>
51     <ul>
52         <li>
53             <a href="/proyecto/presentaciones/p1/" title="Presentación inicial.">
```

```

55         Presentación 1
56     </a>
57 </li>
58 <li>
59     <a href="/proyecto/presentaciones/p2/" title="Segunda presentación.">
60         Presentación 2
61     </a>
62 </li>
63 <li>
64     <a href="/proyecto/presentaciones/p3/" title="Presentación final.">
65         Presentación 3
66     </a>
67 </li>
68 </ul>
69 </div>
70
71 <div style="position: absolute; bottom: 0px; left: 8px;">
72     <a href="https://etseib.upc.edu/" target="_blank" title="Escuela Técnica
73     ↳ Superior de Ingeniería Industrial de Barcelona" style="text-decoration:
74     ↳ none;">
75         
76     </a>
77     <a href="https://upc.edu/" target="_blank" title="Universidad Politécnica
78     ↳ de Cataluña" style="text-decoration: none;">
79         
80     </a>
81 </div>
82 </nav>

```

navbar.html

### Estilos tablero (*tablero.css*)

```

1  #board {
2      border-collapse: collapse;
3      margin: 10px 150px 0 0;
4      float: right;
5      background-color: #D2E0F9;
6  }
7
8  #board td {
9      height: 110px;
10     width: 110px;
11     padding: 0;
12     border: 4px solid Black;
13     text-align: center;
14     font-size: 70px;
15     font-weight: bold;
16 }

```

tablero.css

```
17
18 #X {
19     color: #336600;
20 }
21
22 #O {
23     color: #3862E0;
24 }
25
26 #tic {
27     width: 100px;
28     height: 100px;
29     background-color: #D2E0F9;
30     border: 0px;
31     cursor: pointer;
32 }
33
34 #tic span {
35     text-align: center;
36     font-weight: bold;
37     font-size: 70px;
38 }
39
40 #tic span:after {
41     content: 'O';
42     opacity: 0;
43     transition: 1s;
44 }
45
46 #tic:hover span:after {
47     opacity: .75;
48 }
49
50
51 /* Actualizar con el tamaño. */
52 @media only all
53 and (max-width: 900px) {
54     #board {
55         margin: 0 auto;
56         float: none;
57     }
58 }
59
60 @media only all
61 and (max-width: 1050px)
62 and (min-width: 899px){
63     #board {
64         margin: -15px 50px 0 0;
65     }
66 }
67
68 @media only all
69 and (max-width: 1200px)
```



```
70 and (min-width: 1049px){  
71   #board {  
72     margin: -15px 100px 0 0;  
73   }  
74 }
```

tablero.css