



Universidad Veracruzana

FACULTAD DE INGENIERÍA

DOCTORADO/MAESTRÍA/LICENCIATURA EN INGENIERÍA

TÍTULO DEL PROYECTO DE INVESTIGACIÓN.
TESIS

Tipo de documento, reporte semestral, final, etc.

Alumno: Xxxxx Yxxxx Zxxxx

Asesor: Xxxx Yxxxx Zxxxx

Docente de la asignatura: Dr. Jxxxx Txxxxx Vxxxx
(Opcional)

Septiembre 2077

Índice general

1. Antecedentes	2
1.1. Antecedentes I	2
1.1.1. Caso I	2
2. Técnicas de control	3
2.1. Antecedentes I	3
2.1.1. Caso I	3
3. Técnicas de mitigación	4
3.1. Antecedentes I	4
3.1.1. Caso I	4
4. Retardos de tiempo	5
4.1. Tipos	5
Bibliografía	6
A. Ap. A. Código	8
B. Ap. B. Evidencias	9

Capítulo 1

Antecedentes

1.1. Antecedentes I

1.1.1. Caso I

En este capítulo se trabaja la revisión de antecedentes.

Capítulo 2

Técnicas de control

2.1. Antecedentes I

2.1.1. Caso I

En este capítulo se trabaja la revisión de técnicas de control.

Capítulo 3

Técnicas de mitigación

3.1. Antecedentes I

3.1.1. Caso I

En este capítulo se trabaja la revisión de técnicas de mitigación.

universo. Retardos de tiempo tienen lugar y son perceptibles en diferentes ámbitos de la actividad humana, así como en la naturaleza del universo. [1]

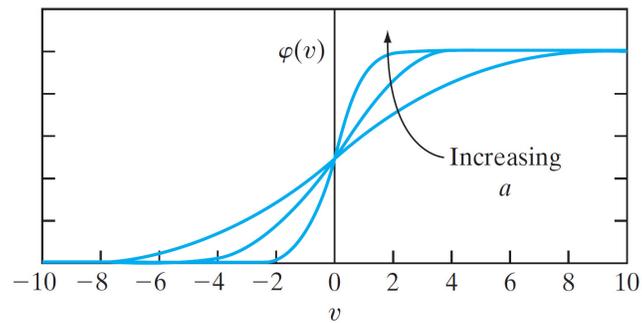


Figura 4.1: Función sigmoide, [2].

- Tiempo.
- Retardo.
- Espacio.

$$\varphi(v) = \begin{cases} 1 & \text{if } v \geq 0 \\ 0 & \text{if } v < 0 \end{cases} \quad (4.1)$$

Ejemplo de estos operadores pueden ser las funciones *AND*, *OR* y *XOR*, de las cuales los valores de sus tablas de verdad se muestran en la tabla 4.1.

x_1	x_2	AND	OR	XOR
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

Tabla 4.1: Tabla verdad función AND, OR y XOR.

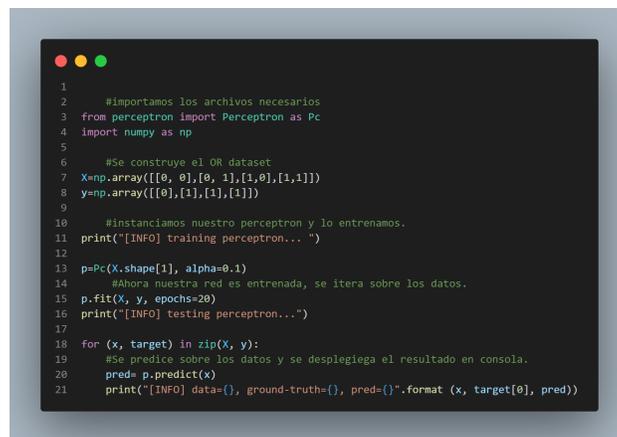
Bibliografía

- [1] R. J. Anderson and M. W. Spong, "Bilateral control of teleoperators with time delay," *IEEE Transactions on Automatic Control*, vol. 34, no. 5, pp. 494–501, 1989.
- [2] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: A Bradford Book, 2018.

Apéndice A

Ap. A. Código

Se anexa un imagen al apéndice A.

A screenshot of a code editor window with a dark background and light-colored text. The code is a Python script for a Perceptron classifier. It includes imports for 'Perceptron' and 'numpy', defines a dataset 'X' and target labels 'y', instantiates a 'Perceptron' object, trains it for 20 epochs, and then iterates over the training data to print predictions and ground truths. The code is numbered from 1 to 21.

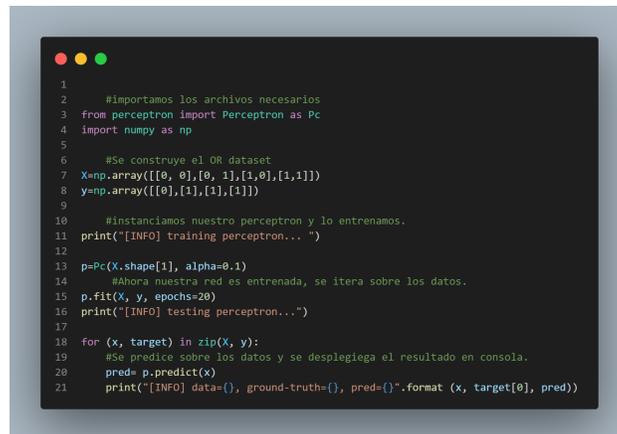
```
1
2 #importamos los archivos necesarios
3 from perceptron import Perceptron as Pc
4 import numpy as np
5
6 #Se construye el OR dataset
7 X=np.array([[0, 0],[0, 1],[1,0],[1,1]])
8 y=np.array([[0],[1],[1],[1]])
9
10 #instanciamos nuestro perceptron y lo entrenamos.
11 print("[INFO] training perceptron... ")
12
13 p=Pc(X.shape[1], alpha=0.1)
14 #Ahora nuestra red es entrenada, se itera sobre los datos.
15 p.fit(X, y, epochs=20)
16 print("[INFO] testing perceptron...")
17
18 for (x, target) in zip(X, y):
19     #Se predice sobre los datos y se despliega el resultado en consola.
20     pred= p.predict(x)
21     print("[INFO] data={}, ground-truth={}, pred={}".format(x, target[0], pred))
```

Figura A.1: script Perceptrón.

Apéndice B

Ap. B. Evidencias

Este anexo incluye evidencia de actividades complementarias. Se anexa un imagen al apéndice B.

A screenshot of a terminal window showing a Python script for a Perceptron model. The script is numbered from 1 to 21. It imports necessary libraries, creates a dataset for an OR gate, trains the perceptron, and tests it on the dataset. The output shows the predicted values for each data point.

```
1
2 #importamos los archivos necesarios
3 from perceptron import Perceptron as Pc
4 import numpy as np
5
6 #Se construye el OR dataset
7 X=np.array([[0, 0],[0, 1],[1,0],[1,1]])
8 y=np.array([0,1,1,1])
9
10 #instanciamos nuestro perceptron y lo entrenamos.
11 print("[INFO] training perceptron... ")
12
13 p=Pc(X.shape[1], alpha=0.1)
14 #Ahora nuestra red es entrenada, se itera sobre los datos.
15 p.fit(X, y, epochs=20)
16 print("[INFO] testing perceptron...")
17
18 for (x, target) in zip(X, y):
19     #Se predice sobre los datos y se despliega el resultado en consola.
20     pred= p.predict(x)
21     print("[INFO] data={}, ground-truth={}, pred={}".format(x, target[0], pred))
```

Figura B.1: script Perceptrón.