# IC-2023: Proyecto

Luis Ballado
luis.ballado@cinvestav.mx

CINVESTAV UNIDAD TAMAULIPAS — April 12, 2023

## 1 Instrucciones para ejecución

**Info:** Se adjunta la liga al repositorio, donde se encuentran los códigos ver código en github

**Info:** Se hacen uso de las siguientes bibliotecas de Python

- argparse - creación de banderas para alimentar el programa
- fuzzylogic - construcción de Lógica Borrosa
- traci - Interfaz de SUMO en su versión de python
- tabulate - generación de tablas en línea de comandos
- sumolib - Bibliotecas SUMO

1. Instalar Simulador SUMO en una distro LINUX

Command Line

```
$ sudo add-apt-repository ppa:sumo/stable
$ sudo apt-get update
$ sudo apt-get install sumo sumo-tools sumo-doc
```

2. Instalar dependencias con el archivo requirements.txt incluido en los archivos. (python >= 3)

Command Line

```
$ pip install -r requirements.txt
```

3. Clonar el repositorio

Command Line

```
$ git clone https://github.com/luisballado/InteligenciaComputacional.git
$ cd InteligenciaComputacional
$ cd proyecto
$ cd fuzzy_semaphore
```

4. Descripción Archivos

   Dentro del repositorio se sigue la siguiente estructura para organizar los archivos del proyecto (archivos xml que definen los parámetros del mapa a construir en el simulador SUMO):

```
fuzzy_semaphore/
├── logica_borrosa.py
├── requirements.txt
├── sumo_run.py
├── victoria_cluster.add.xml
├── victoria_cluster.net.xml
├── victoria_cluster.net.xml.gz
├── victoria_cluster.neteditcfg
├── victoria_cluster.rou.xml
├── victoria_cluster.sumocfg
├── victoria_cluster_ligero.rou.xml
├── victoria_cluster_medio.rou.xml
└── victoria_cluster_pesado.rou.xml
```

- **logica_borrosa.py** - Clase Lógica Borrosa donde se hacen los cálculos
- **sumo_run.py** - Programa principal para ejecutar el simulador con el mapa de CD. Victoria.
- **victoria_cluster.** - Archivos generados por el Wizard al construir el mapa seleccionado

# 2 Valores de los Parámetros

En la ejecución del programa se incluyen banderas para su ejecución con Interfaz gráfica o no, y también la cantidad de trafico a generar.

- –show True ó –show False
- –traffic Bajo ó –traffic Medio ó –traffic Alto

Dependiendo de la versión de python en su máquina se correria de la siguiente forma:

Command Line

```
$ python3 sumo_run.py --show True --traffic Bajo
```

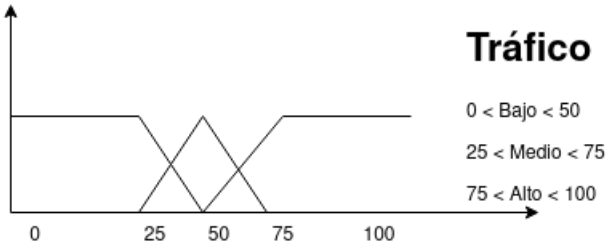
## 2.1 Lógica Difusa

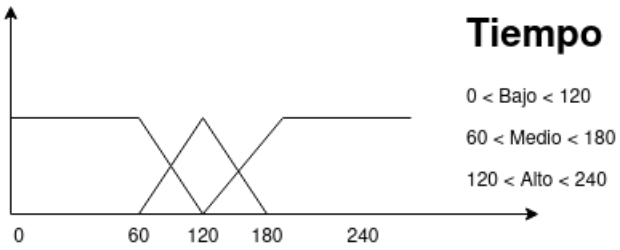
Se propone un control de lógica difusa para el control de los semaforos, partiendo de la premisa que podemos contar la cantidad de carros gracias al sensor lane area detector, así también contabilizar el tiempo de su última participación en el cluster.

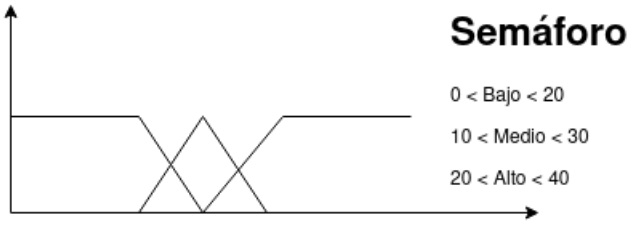
### 2.1.1 Dominio

- Trafico : [0 - 100]
- Tiempo : [0 - 240]
- Tiempo Semaforo : [0 - 60]

### 2.1.2 Conjuntos



# 3 Estadísticas

Figure 1: Gráfica % llenado

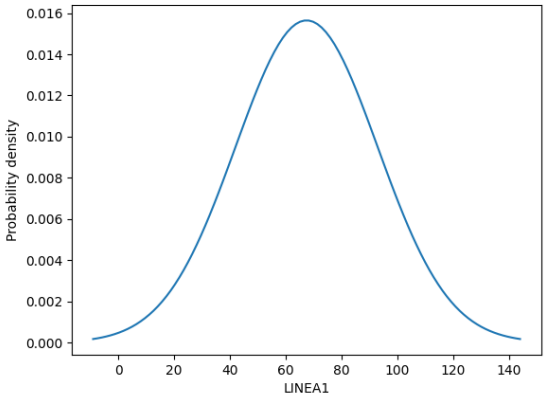



Figure 2: Gráfica tiempo

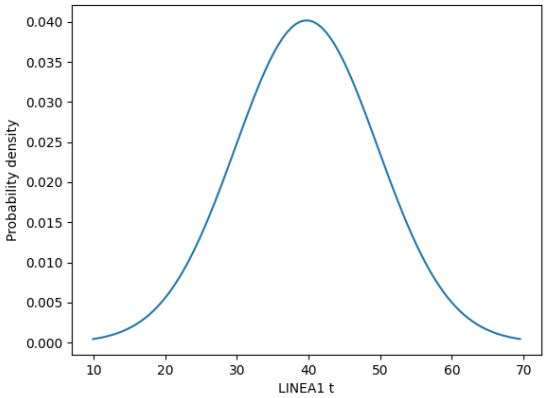



Figure 3: Gráfica % llenado

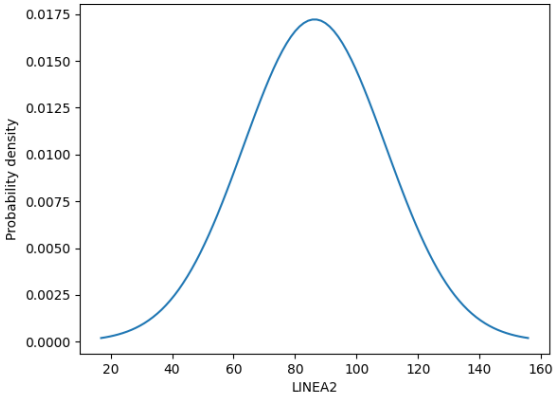


Figure 4: Gráfica tiempo

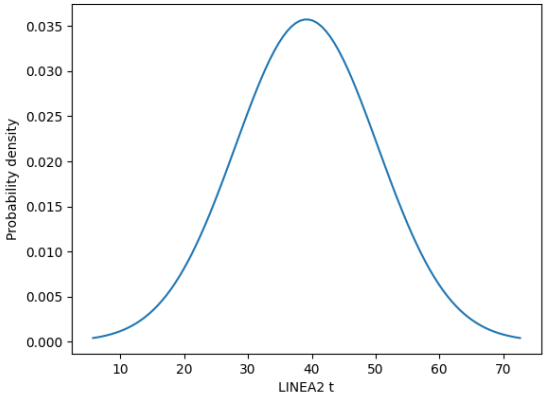



Figure 5: Gráfica % llenado

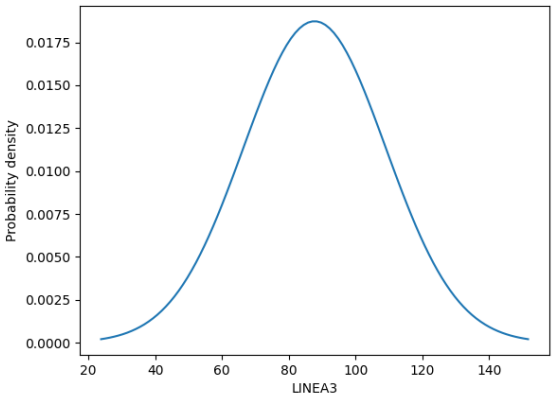
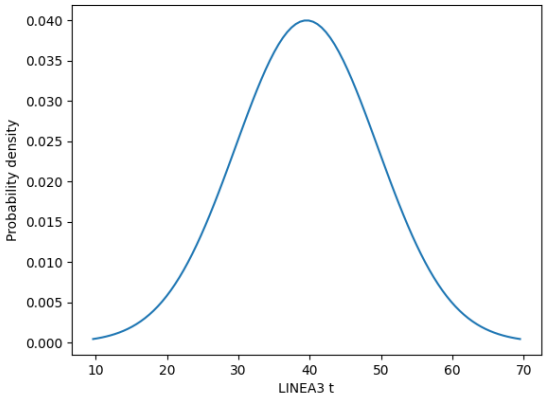



Figure 6: Gráfica tiempo




# 4 Referencias

Traffic Lights Control
Interfaz SUMO-Python
Ejemplo Manejo Interfaz Python-SUMO
Lane Area Detector

Figure 7: Gráfica % llenado

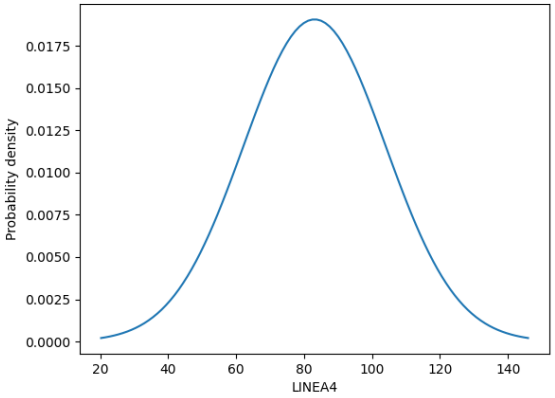
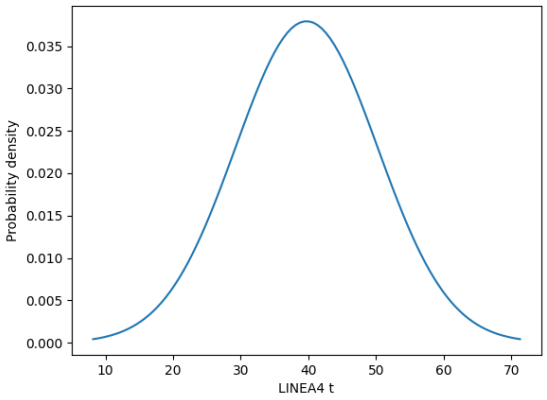



Figure 8: Gráfica tiempo




Tutorial SUMO - A Road Map SUMO