

## NetSim Mobility File Generation

**Software:** NetSim Academic/Standard/Pro, Python 3.11

### Project Download Link:

[https://github.com/NetSim-TETCOS/Mobility\\_File\\_Generator\\_v14.0/archive/refs/heads/main.zip](https://github.com/NetSim-TETCOS/Mobility_File_Generator_v14.0/archive/refs/heads/main.zip)

### Introduction

Mobility python script allows user to automatically generate the mobility.csv file without the need for manually writing from GUI. Mobility python script meets the user requirements which involves changing the value of time and distance co-ordinates.

### Variables used in the python script for generating mobility input to NetSim.

1. **mobility\_type** – This is used to get either linear or circular form of mobility file.
2. **n** - Total number of devices in which file-based mobility is enabled.
3. **deviceid** – list of device id's that user wants to add into the mobility file separated by comma(,).
4. **velocity** - Velocity at which device will move from one point to another.
5. **x\_coordinates** –Initial position of the device.
6. **y\_coordinates** –Initial position of the device.
7. **distanceStep** – It is used in finding the next location for each device.
8. **timePerStep** – Time gap between two locations.
9. **timeStep** – It is used along with timePerStep to get the time value for node to move.
10. **lengthOfSteps** - Range in which, time value will add into mobility file.

### How to generate Mobility.csv file

#### Case 1: Mobility file for uniform straight-line motion

1. Open the Mobility-File-Generator.py script in python IDE.
2. Users can modify parameters like mobility\_type, n, distanceStep, timePerStep, timeStep, velocity and x, y coordinates.

```

1 # Import necessary libraries
2 import math
3 import random
4 import tkinter as tk
5 from tkinter import messagebox
6
7 # Initialize empty lists to store mobility data and device properties
8 array = [] # List to store mobility data
9 vel = [] # List to store velocities of devices
10 x = [] # List to store x-coordinates of devices
11 y = [] # List to store y-coordinates of devices
12
13 # Define the type of mobility: 'linear' or 'circular'
14 mobility_type = "linear" # Set mobility_type = "circular" for circular motion
15
16 # Total number of devices
17 n = 3
18
19 # List of device IDs (1-indexed) present in the file
20 deviceid = [1, 3, 5]
21
22 # Define steps in distance and time
23 # Steps in distance device has to move based on velocity
24 # Steps in distance = velocity * distanceStep
25 distanceStep = 1
26
27 # Time gap between two locations/positions
28 # time(s) = lengthOfSteps / timeStep, formula to calculate the next position time
29 timeStep = 5.0
30
31 timePerStep = 3 # Time per step
32
33 # Define the range at which time will be added into mobility.txt file
34 lengthOfSteps = 1804
35
36 # Define the velocity at which devices are moving
37 velocity = 3.0
38
39 # Define the initial position of devices
40 x_coordinates = 280.0
41 y_coordinates = 400.0
42
43 # Define the boundaries of the grid
44 grid_min_x = 0.0
45 grid_min_y = 0.0
46 grid_max_x = 1000.0
    
```

Figure 1: User Modify the parameters like velocity and x, y coordinates in Mobility-File-Generator.py.

3. Generating Mobility file is started by opening command prompt in the directory of the Mobility script project and starting the python script as shown below.

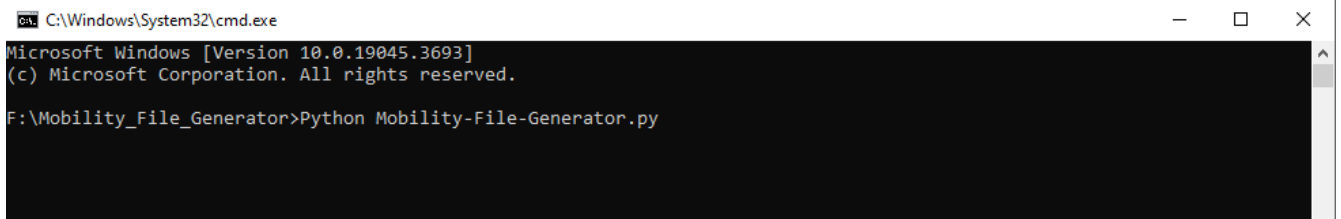
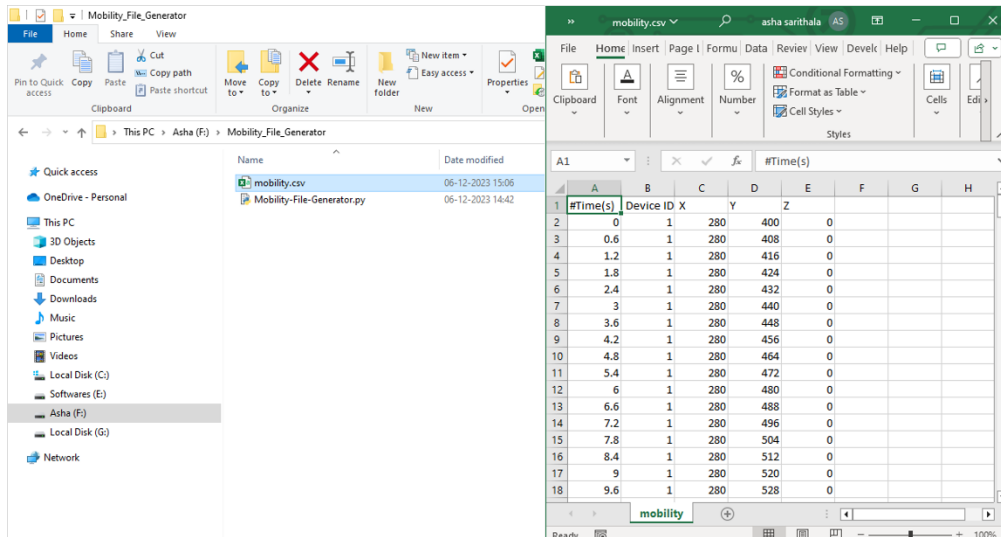


Figure 2: Generating Mobility file using command prompt.

4. After executing the command mobility.csv file will be created in the same folder that contains the Mobility-File-Generator.py python script.
5. A sample uniform straight line mobility file generated using this script is shown below.



**Figure 3:** Mobility file generated using the script.

## Case 2: Mobility file for uniform circular motion

1. Users need to set `mobility_type = "Circular"` and other steps are similar as shown below.

```

1 # Import necessary libraries
2 import math
3 import random
4 import tkinter as tk
5 from tkinter import messagebox
6
7 # Initialize empty lists to store mobility data and device properties
8 array = [] # List to store mobility data
9 vel = [] # List to store velocities of devices
10 x = [] # List to store x-coordinates of devices
11 y = [] # List to store y-coordinates of devices
12
13 # Define the type of mobility: 'linear' or 'circular'
14 mobility_type = "Circular" # Set mobility_type = "circular" for circular motion
15
16 # Total number of devices
17 n = 3
18
19 # List of device IDs (1-indexed) present in the file
20 deviceid = [1, 3, 5]
21
22 # Define steps in distance and time
23 # Steps in distance device has to move based on velocity
24 # Steps in distance = velocity * distanceStep
25 distanceStep = 1
26

```

**Figure 4:** Modify the Mobility Type to Circular.

2. A sample uniform circular mobility file generated using this script is shown below.

The screenshot shows a file explorer window displaying the 'mobility.csv' file. To the right, an Excel spreadsheet is open, showing the data generated by the script. The spreadsheet has columns for Time (s), Device ID, X, Y, and Z. The data shows a circular path for Device ID 1 over 18 time steps.

#Time(s)	Device ID	X	Y	Z
0	1	280	400	0
0.6	1	275.5	406.6	0
1.2	1	267.5	406.4	0
1.8	1	263.4	399.5	0
2.4	1	266.9	392.4	0
3	1	274.8	391.5	0
3.6	1	279.9	397.6	0
4.2	1	277.4	405.3	0
4.8	1	269.7	407.3	0
5.4	1	263.8	401.9	0
6	1	265.2	394	0
6.6	1	272.6	390.9	0
7.2	1	279.1	395.4	0
7.8	1	278.9	403.4	0
8.4	1	272	407.5	0
9	1	264.9	404	0
9.6	1	264	396	0

**Figure 5:** Circular mobility file generated using the script.

The script can be modified to generate mobility patterns in addition to those that are supported currently this script supports Linear and Circular mobility types.