b10505005_hw3_readme.pdf

Description

Use the file to plot figures of describing handoffs during mobile devise using

- Arranging cells ID
- output the list all the time when the handoff event occurs (coordination base)
- Calculate total handoff during simulation time

Requirement

Python 3.8.10

Main Function

a. basic plot of the coordinates of base station 1-19

```
# Define parameters
min_speed = 1  # Minimum speed (m/s)
max_speed = 15  # Maximum speed (m/s)
min_t = 1  # Minimum travel time (s)
max_t = 6  # Maximum travel time (s)
total_time = 900  # Total simulation time (s)
hex_radius = 500  # Radius of hexagonal cell (m)
```

```
def in_hexagon(x, y, radius):

    if np.abs(y) > radius * (np.sqrt(3)/2) and np.abs(x) <
0.5*radius:
        return False
    if np.abs(x) > radius - np.abs(y) * (1/np.sqrt(3)):
        return False
    else:
        return True
```

```
def hexagon_vertices(side_length):
    return [
        (side_length * np.cos(np.pi / 3 * i), side_length *
np.sin(np.pi / 3 * i))
        for i in range(6)
    ]
```

```
def hexagonal_grid(ISD):
    coords = [(0, 0)]  # 中心點
    for i in range(6):
        angle = -np.pi / 6 + (np.pi / 3 * i)
        x = np.cos(angle) * ISD
        y = np.sin(angle) * ISD
        coords.append((x, y))

    for i in range(6):
        angle = (np.pi / 3 * i)
        x = np.cos(angle) * ISD * np.sqrt(3)
        y = np.sin(angle) * ISD * np.sqrt(3)
        coords.append((x, y))

    for i in range(6):
        angle = (-np.pi / 6 + np.pi / 3 * (i + 1))
        x = np.cos(angle) * 1000
        y = np.sin(angle) * 1000
        coords.append((x, y))

    return np.array(coords)
```

b. oordination-based, record the movwmwnt of mobile device

```
def get_current_cell(x, y, cells):
    distances = [np.sqrt((x - cx) ** 2 + (y - cy) ** 2) for
(cx, cy) in cells]
    return np.argmin(distances) + 1
```

```
time = 0
x, y = 250, 0
current_cell = get_current_cell(x, y, cells)
handoff = 0

moving_x = []
moving_y = []

while time < total_time:
    direction = random.uniform(0, 2 * np.pi)
    speed = random.uniform(min_speed, max_speed)
    travel_time = random.uniform(min_t, max_t)

    dx = speed * np.cos(direction) * travel_time
    dy = speed * np.sin(direction) * travel_time
    x += dx
    y += dy
    moving_x.append(x)
    moving_y.append(y)
    time += travel_time

    new_cell = get_current_cell(x, y, cells)

    if new_cell != current_cell:
        handoff_log.append(f"{int(time)}s {current_cell} {n
ew_cell}")
        handoff += 1
        current_cell = new_cell  # Update to the new cell
```

Compile

```
  python 3-1.py
  python 3,2.py
```

Contact：ESOE b10505005 蔣依倢