Point projection

Map Matching Project Meeting

@서울시립대. 2020. 7. 16.

시립대 한양대 부족한 인생

1:1 matching된 택시 경로와 도로 segment가 주어졌을 때, 택시 경로의 각 GPS가 어느 도로에 가까운지 추측하는 과정

점과 선 사이의 거리 구하기

이교헌 선생님께서 작성해주신 코드 이용

```python
def distance_line_point_new(line, point):
    """
    calculate distance between point and each line

    Parmeter
    --------
    line : np.array
           ex) ([[x1,y1],[x2,y2],[x3,y3]])
    point : np.array([x,y])
    """
    lines = line[:-1] - line[1:]
    a = -lines[:,1]
    b = lines[:,0]
    c = -a * line[:,0][:-1] - b * line[:,1][:-1]
    shortest = np.abs(a*point[0]+b*point[1]+c)/np.sqrt(a*a+b*b)
    m1=-a/(b+1e-12)
    m2=-1/(m1+1e-12)

    x = (m1*line[:,0][:-1]-m2*point[0]-line[:,1][:-1]+point[1])/(m1-m2)
    y = m2*(x-point[0])+point[1]

    yesorno = (line[:,0][:-1]-x)*(line[:,0][1:]-x) + (line[:,1][:-1]-y)*(line[:,1][1:]-y)

    len1 = np.sqrt((line[:,0][:-1]-point[0])**2+(line[:,1][:-1]-point[1])**2)
    len2 = np.sqrt((line[:,0][1:]-point[0])**2+(line[:,1][1:]-point[1])**2)
    short = shortest*(yesorno<=0)+np.minimum(len1,len2)*(yesorno>0)

    return short
```

- 도로 segment(선)와 택시 GPS(점)의 거리를 구할 때 이용
- 이를 위해 기존 stitching과정을 거친 segment을 선 형태로 고쳐야함
- 함수에 들어가는 선 형태 : 좌표 list [[x1, y1], [x2, y2], …]

도로 segment의 각 노드들의 위치좌표 구하기

```python
def segment_to_line(G, segment):# G -> self(Network)
    """
    segment -> pos_array [x_node,y_ndoe]

    Parameter
    ----------
    G : road network
    segment


    Return
    -------
    pos_array
    np.array([x1,y1],[x2,y2],...)
    """
    pos_list = np.zeros([len(segment.nodes()),2])
    for c in range(len(segment.nodes())):
        pos_list[c] = G.nodes[segment.nodes()[c]]['pos']
    return pos_list
```

segment

network

```
segment_to_line(Seoul, seg1)
```

```
array([[ 327560.3864, 4149082.139 ],
       [ 327541.7048, 4149106.248 ],
       [ 327574.0778, 4149130.621 ],
       [ 327610.2074, 4149157.824 ],
       [ 327636.432 , 4149178.394 ],
       [ 327660.9839, 4149197.656 ],
       [ 327683.5034, 4149215.34  ],
       [ 327709.2063, 4149235.51  ],
       [ 327735.2327, 4149255.93  ],
       [ 327759.5504, 4149275.019 ],
       [ 327752.4094, 4149285.213 ],
       [ 327791.4538, 4149297.282 ],
       [ 327823.9695, 4149317.079 ],
       [ 327848.2711, 4149335.403 ],
       [ 327871.0986, 4149352.614 ],
       [ 327895.0851, 4149370.69  ],
       [ 327917.2833, 4149387.426 ],
       [ 327934.0532, 4149400.067 ]])
```

- 도로 네트워크와 segment를 같이 입력 받음
- segment의 각 노드들의 위치 데이터를 찾아 return

도로 segment의 각 노드들의 위치좌표 구하기

```python
def point_projection(G,target, path):# G, target -> self
    """
    Parameter
    ---------
    target : np.array? pos array?
    path : segment

    Return
    ------
    edge_list : list
        taxi trajectory passed
    """
    edge_list = []
    path_line = segment_to_line(G,path)
    for target_point in target:
        distance_list = distance_line_point_new(path_line, target_point)
        edge_list.append(path.edges()[np.where(min(distance_list)==distance_list)[0][0
    return edge_list
```

```
projection_edges
```

```
[(102960, 103166, 0),
 (102960, 103166, 0),
 (103166, 103386, 0),
 (103602, 103798, 0),
 (103798, 103990, 0),
 (103990, 104180, 0),
 (103990, 104180, 0),
 (104414, 104614, 0),
 (104614, 104858, 0),
 (104614, 104858, 0),
 (104614, 104858, 0),
 (104972, 105093, 0),
 (105289, 105521, 0),
 (105289, 105521, 0),
 (105695, 105877, 0),
 (105877, 106039, 0),
 (105877, 106039, 0),
 (106039, 106193, 0)]
```

```
seg1.edges()
```

```
[(102960, 103166, 0),
 (103166, 103386, 0),
 (103386, 103602, 0),
 (103602, 103798, 0),
 (103798, 103990, 0),
 (103990, 104180, 0),
 (104180, 104414, 0),
 (104414, 104614, 0),
 (104614, 104858, 0),
 (104858, 104972, 0),
 (104972, 105093, 0),
 (105093, 105289, 0),
 (105289, 105521, 0),
 (105521, 105695, 0),
 (105695, 105877, 0),
 (105877, 106039, 0),
 (106039, 106193, 0)]
```

```python
%%timeit
projection_edges = point_projection(Seoul, target_test, seg1)
```

```
1.71 ms ± 12.1 µs per loop (mean ± std. dev. of 7 runs, 1000 loops each)
```