По сути решили, что dist проще считать таким образом. Взяли прямую, относительно которой будем откладывать новую. Такая находится сейчас в find\_nearest\_line. От этой прямой нашли точку излома наиболее близкую к центру. Центр можно хардкодить, т.к. графики фиксированные. Берём соседнюю ломаную и перебираем расстояние от этой точки до всех её отрезков. Нужно проверять принадлежность именно отрезку, тут алгоритм надо будет продумать. Далее это расстояние нужно пропорционально collection\_point откладывать.

Нужно найти ближайшую к центру точку излома. Функция принимает прямую line и аргумент middle\_point. следующей структуры. {‘collection\_point’: 120, 'start': (61.5, 372), 'points': [(79.5, 427)], 'end': (89, 470)}. Нужно сложить в массив new\_points точку start, все точки из массива points у line и точку end. Перебираем получившийся массив new\_points и сверяем координату x каждой точки с числом middle\_point, результатом работы функции будет координата x, которая ближе всего лежит к middle\_point.

После нахождения точки, надо определить, между каких прямых будет лежать наша ломаная. По сути entrance\_collection\_point лежит между двух прямых. Тут возможны следующие случаи entrance\_collection\_point > collection\_point у found\_line или entrance\_collection\_point < collection\_point у found\_line.

entrance\_collection\_point > collection\_point: нужно взять прямую выше found\_line. Т.е. больше с минимальной разностью.

entrance\_collection\_point < collection\_point: нужно взять прямую ниже found\_line. Т.е. меньше с минимальной разностью.

Нужна функция. Нахождение второй прямой. Аргументы функции entrance\_collection\_point, found\_line и lines. line – словарь следующей структуры {‘collection\_point’: 120, 'start': (61.5, 372), 'points': [(79.5, 427)], 'end': (89, 470)}. Lines- массив словарей структуры описанной выше. Функция делает следующий алгоритм, если entrance\_collection\_point > line[‘collection\_point’]. Тогда нужно перебрать все lines и взять из них такую, collection\_point которой больше чем у line, причём разница между этими значениями минимальна. Если entrance\_collection\_point < line[‘collection\_point’], тогда нужно перебрать все lines и взять из них такую, collection\_point которой меньше, чем у line, причём разница между этими значениями минимальна

Чтобы при расчётах учитывалась верхняя ломаная и нижняя их можно просто положить в массив, но по ним ломаные не строить

Нужна функция, которая находит расстояние от точки до отрезка. Открытая проблема, т.к. всё считается криво.

Расстояние от точки до ломаной. Нужна функция, которая принимает два аргумента line и point. Point – точка с координатами x и y. line – словарь следующей структуры {‘collection\_point’: 120, 'start': (61.5, 372), 'points': [(79.5, 427)], 'end': (89, 470)}. Нужно сложить в массив points точки start, все точки из line[‘points’] и точку end. Далее нужно перебрать этот массив, нужно брать пары точек и передавать их в функцию distance\_to\_line(point[i], point[i+1], nearest\_point), третий аргумент nearest\_point из глобальной области видимости.