**ОБЩИЙ ПОДХОД К РЕШЕНИЮ**

Подобные задачи целесообразно декомпозировать и для каждого декомпозированного содержания определить концептуальный метод решения. Понятно, что генерация координатных точек, изображающих некую ломаную траекторию обеспечивает лишь модель данных, которые скармливаются разрабатываемому фильтру прямых линий (SLF), без них его в принципе не удастся тестировать и отлаживать. Можно, конечно, взять лидар или т.п. готовое устройство, где такие данные являются выходными, но в настоящем случае это технически «дорого» - сымитировать ломаную, окружность и другие формы траекторий точками будь то под шумом или нет легко программно (Python):

def firstPnt(pnts : np.ndarray) -> None:

    pnts[0, 0] = 0.5 \* random.rand() - 0.25

    pnts[1, 0] = 0.5 \* random.rand() - 0.25

    pnts[2, 0] = 2.0 \* m.pi \* random.rand() - m.pi

def createPnts(pnts : np.ndarray, N, d0 = 0.1, shape = 0, mess = 0.1) -> None:

    i\_ang = 0

    deltaAng = 0.2 \* random.rand() - 0.

    for i in range(1, N):

        d = d0 \* (1 + random.randn() \* mess)

        pnts[0, i] = pnts[0, i - 1] + d \* m.cos(pnts[2, i - 1])

        pnts[1, i] = pnts[1, i - 1] + d \* m.sin(pnts[2, i - 1])

        if (shape == 0):    #polyline

            if (random.rand() > 1 - 5.0 / N): # 5 fractures in average

                pnts[2, i] = pnts[2, i - 1] + m.pi \* random.rand() - m.pi/2

                i\_ang = i

            else:

                pnts[2, i] = pnts[2, i\_ang] \* (1 + random.randn() \* mess)

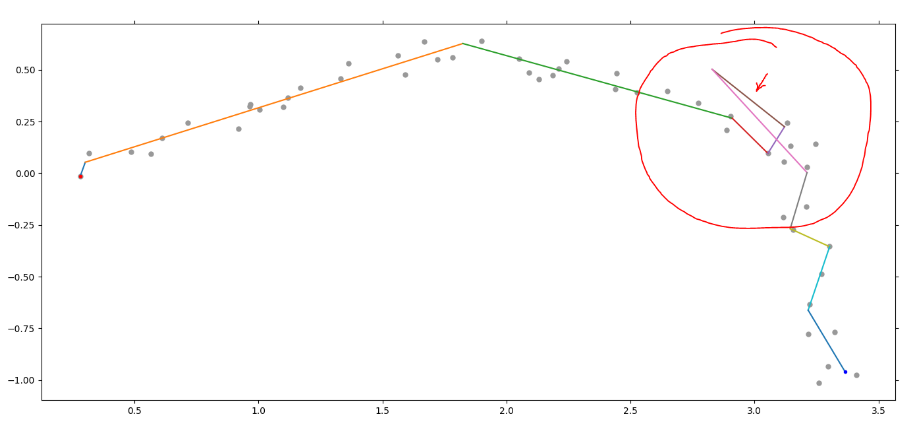
        elif (shape == 1):  #circle

            pnts[2, i] = pnts[2, i - 1] + deltaAng

*#первую точку можно и не отделять явно*

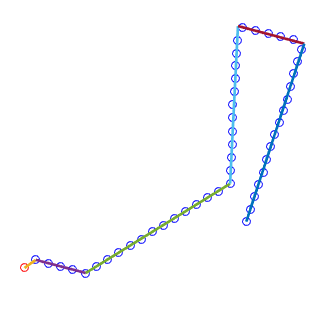
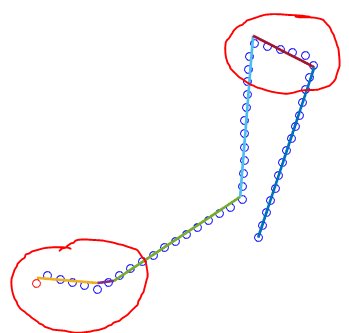
Сам SLF распадается на первой очереди на 2 содержания: каноническую задачу линейной регрессии, т.е. приближения скопления точек прямой (англ. line fitting), и сегментацию скопления, что вместе можно назвать просто кусочно-линейной регрессией (англ. piecewise linear regression) – легко гуглится и можно смотреть, что там предлагается по этому вопросу.

Основная же сложность всего задания лежит во второй очереди, когда запрошено дать сегментацию именно непрерывной ломаной без откровенных вылетов. Вот пример такого вылета:



Дело в том, что в первоначально полученном наборе прямых, приближающих сегменты, две соседние прямые могут пересекаться в точке, расположенной хоть бесконечно далеко от всего скопления (когда они окажутся параллельными).

При дальнейшем углублении проявляется трудность, связанная с поиском лучших вариантов сегментации при сохранении допуска на вписывание прямых. Так, например, критерием обнаружения границы сегмента может выступать превышение расстояния от следующей точки до предполагаемой прямой, построенной по предыдущим точкам, над некоторым порогом (это по замыслу и есть tolerance) – но в таком случае в сегмент скопления, приближенный отдельной прямой, могут попасть и точки, которые по факту начинают уже следующий сегмент исходной ломаной:



*SLF* *с и без коррекции границ сегментов (все остальные параметры идентичны)*

Кстати, к этой трудности в задании не было ни отсылок, ни упоминаний, а в видеоиллюстрации она пропущена. Мы посчитали, что с одной стороны это переусложнение, а с другой, самостоятельно её заметивший – красавчик.