```
274
275         if endPoint != None:
276             if currentLocation.x <= endPoint.x and currentLocation.y <= endPoint.y:
277                 break
278         # reset criticalPoints every 1000 STEPs
279         criticalPointsOfClass1 = sorted(class1.points, key=currentLocation.distanceFro
280         criticalPointsOfClass2 = sorted(class2.points, key=currentLocation.distanceFro
281         print('passing loop, current point:', currentLocation.rawValue, 'dir:', lastDi
282
283     with open(fileName, fileMode) as file:
284         csvFile = csv.writer(file, delimiter=' ')
285         csvFile.writerows(recogLinePoints)
286
287
288 def moveToNextLocation(currentLocation, lastDirection, cPoints1, cPoints2):
289     minDifferences = []
290     for direction in list(validDirections.keys()):
291         if validDirections[lastDirection][0] == -validDirections[direction][0] and val
292             continue
293
294         minDistanceOfClass1 = sorted([ point.distanceFromXYPoint(currentLocation.move(
295         minDistanceOfClass2 = sorted([ point.distanceFromXYPoint(currentLocation.move(
296         difference = {'distance': abs(minDistanceOfClass1 - minDistanceOfClass2), 'dir
297         minDifferences.append(difference)
298
299     bestPoint = sorted(minDifferences, key=lambda difference: difference['distance'])[
300     nextDirection = bestPoint['direction']
301     nextLocation = currentLocation.move(nextDirection)
302     isRecogLinePoint =  bestPoint['distance'] <= DELTA
303
304     return (nextLocation, nextDirection, isRecogLinePoint)
305
306
307
308 # NOTE: - plot results:  ********************************************************  plot
309
310 def plotResultOf(result, class1, class2, testData, title, recogLine=False):
311
312     # MARK: - plot learned data
313     # note that all learned data are represented by color gray with different marker t
314     pl.plot(class1.toNdarray[:,0], class1.toNdarray[:,1], 'o', color='tab:gray', label
315     pl.plot(class1.mean[0], class1.mean[1], 'o', color='k', markersize=8)
316     pl.plot(class2.toNdarray[:,0], class2.toNdarray[:,1], '+', color='tab:gray', label
317     pl.plot(class2.mean[0], class2.mean[1], '+', color='k', markersize=10)
318
```