

spiegelwuerfel.py

```
001 # Überprüfung der Rechtwinkligkeit
002 # #####
003
004 # Authors:
005 # Joshua Wolf
006 # Silas Teske
007 # Lasse Zeh
008 # Christopher Mahn
009
010 # #####
011
012 # Import of Libraries
013 # -----
014
015 # import math as m
016 # import string as st
017 # import random as r
018 # import re
019 import numpy as np
020 import os
021
022
023 # -----
024 # Debugging-Settings
025
026 verbose = True # Shows more debugging information
027
028
029 # Functions
030 # -----
031
032
033 # Classes
034 # -----
035
036
037 # Beginning of the Programm
038 # -----
039
040 if __name__ == '__main__':
041
042     # Import der Messwerte des Neigungssensors
043     file = open(os.path.join("data", "imu_data_converted.txt"))
044     imu_data = file.readlines()
045     file.close()
046     for i, e in enumerate(imu_data):
047         imu_data[i] = e.strip().split(";")
048         temp = []
049         for j in imu_data[i]:
050             temp.append(float(j))
051         imu_data[i] = temp
052
053     # Import der Messwerte des Tachymeters
054     file = open(os.path.join("data", "totalstation_data_converted.txt"))
055     totalstation_data = file.readlines()
056     file.close()
057     for i, e in enumerate(totalstation_data):
058         totalstation_data[i] = e.strip().split(";")
059         temp = []
060         for j, f in enumerate(totalstation_data[i]):
061             if(j != 1):
062                 temp.append(float(f))
063             else:
064                 temp.append(f)
065         totalstation_data[i] = temp
066
067     # Berechnung der Orientierung der IMU
068     ori1IMU = []
069     ori1IMU.append(np.average([imu_data[0][8], imu_data[1][8], imu_data[2][8]]))
070     ori1IMU.append(np.average([imu_data[0][9], imu_data[1][9], imu_data[2][9]]))
071     ori1IMU.append(np.average([imu_data[0][10], imu_data[1][10], imu_data[2][10]]))
072     if(verbose):
073         print(f"[Debug] ori1IMU: {ori1IMU}")
074
075     ori2IMU = []
076     ori2IMU.append(np.average([imu_data[7][8], imu_data[8][8], imu_data[9][8]]))
077     ori2IMU.append(np.average([imu_data[7][9], imu_data[8][9], imu_data[9][9]]))
078     ori2IMU.append(np.average([imu_data[7][10], imu_data[8][10], imu_data[9][10]]))
079     if(verbose):
080         print(f"[Debug] ori2IMU: {ori2IMU}")
```

```

081
082 avg_imu1 = []
083 avg_imu1.append(np.average([totalstation_data[1][3],totalstation_data[3][3],totalstation_data[5]
    [3]]))
084 avg_imu1.append(np.average([totalstation_data[1][4],totalstation_data[3][4],totalstation_data[5]
    [4]]))
085 if(verbose):
086     print(f"[Debug] avg_imu1: {avg_imu1}")
087
088 avg_imu2 = []
089 avg_imu2.append(np.average([totalstation_data[7][3],totalstation_data[9][3],totalstation_data[11]
    [3]]))
090 avg_imu2.append(np.average([totalstation_data[7][4],totalstation_data[9][4],totalstation_data[11]
    [4]]))
091 if(verbose):
092     print(f"[Debug] avg_imu2: {avg_imu2}")
093
094 avg_mirror1 = []
095 avg_mirror1.append(np.average([totalstation_data[0][3],totalstation_data[2][3],totalstation_data[4]
    [3]]))
096 avg_mirror1.append(np.average([totalstation_data[0][4],totalstation_data[2][4],totalstation_data[4]
    [4]]))
097 if(verbose):
098     print(f"[Debug] avg_mirror1: {avg_mirror1}")
099
100 avg_mirror2 = []
101 avg_mirror2.append(np.average([totalstation_data[6][3],totalstation_data[8]
    [3],totalstation_data[10][3]]))
102 avg_mirror2.append(np.average([totalstation_data[6][4],totalstation_data[8]
    [4],totalstation_data[10][4]]))
103 if(verbose):
104     print(f"[Debug] avg_mirror2: {avg_mirror2}")
105
106 orilmirror = []
107 orilmirror.append(orilIMU[0])
108 orilmirror.append((avg_imu1[1]-avg_mirror1[1])+orilIMU[1]) # roll
109 orilmirror.append((avg_imu1[0]-avg_mirror1[0])+orilIMU[2]) # pitch
110 if(verbose):
111     print(f"[Debug] orilmirror: {orilmirror}")
112
113 ori2mirror = []
114 ori2mirror.append(ori2IMU[0])
115 ori2mirror.append((avg_imu2[1]-avg_mirror2[1])+ori2IMU[1]) # roll
116 ori2mirror.append((avg_imu2[0]-avg_mirror2[0])+ori2IMU[2]) # pitch
117 if(verbose):
118     print(f"[Debug] ori2mirror: {ori2mirror}")
119
120 difference = []
121 difference.append(ori2mirror[0]-orilmirror[0])
122 difference.append(ori2mirror[1]-orilmirror[1])
123 difference.append(ori2mirror[2]-orilmirror[2])
124 if(verbose):
125     print(f"[Debug] difference: {difference}")
126
127 differencegon = []
128 differencegon.append(difference[0]*(200/np.pi))
129 differencegon.append(difference[1]*(200/np.pi))
130 differencegon.append(difference[2]*(200/np.pi))
131 if(verbose):
132     print(f"[Debug] differencegon: {differencegon}")
133
134 # Berechnungskontrolle
135 innenwinkel_imu = -(orilIMU[2] - ori2IMU[2])
136 drehung_totalstation1 = avg_mirror1[0] - avg_imu1[0]
137 drehung_totalstation2 = avg_imu2[0] - avg_mirror2[0]
138 innenwinkel = 2*np.pi - innenwinkel_imu - drehung_totalstation1 - drehung_totalstation2
139 if(verbose):
140     print(f"[Debug] Innenwinkel (IMU): {(innenwinkel_imu/np.pi*200):.5f} gon")
141     print(f"[Debug] Innenwinkel (Totalstation1): {(drehung_totalstation1/np.pi*200):.5f} gon")
142     print(f"[Debug] Innenwinkel (Totalstation2): {(drehung_totalstation2/np.pi*200):.5f} gon")
143     print(f"[Debug] Innenwinkel: {(innenwinkel/np.pi*200):.5f} gon")
144
145 # Export results
146 file = open(os.path.join("data","results.txt"),"w")
147 file.writelines(f"Result\n")
148 file.writelines(f"-----\n\n")
149 file.writelines(f"Rotation of the mirror (Roll, Pitch, Yaw): {differencegon}\n\n")
150 file.writelines(f"Other Values:\n")
151 file.writelines(f"orilIMU: {orilIMU}\n")
152 file.writelines(f"ori2IMU: {ori2IMU}\n")
153 file.writelines(f"avg_imu1: {avg_imu1}\n")
154 file.writelines(f"avg_imu2: {avg_imu2}\n")
155 file.writelines(f"avg_mirror1: {avg_mirror1}\n")

```

```
156     file.writelines(f"avg_mirror2: {avg_mirror2}\n")
157     file.writelines(f"ori1mirror: {ori1mirror}\n")
158     file.writelines(f"ori2mirror: {ori2mirror}\n")
159     file.close()
```