International Institute of Information Technology Bangalore

Signal processing

Signal processing project

Name of the student: N.V.S.Asrith (IMT2021509)

Name of the student: Rohith kumar (IMT2021504)

Name of the student: Anshu biswas (IMT2021534)

April 29, 2023

0.1 Introduction

GPS (Global Positioning System) and IMU (Inertial Measurement Unit) are two different technologies used for determining the position of an object or a person. While both technologies can be used for distance calculation, they work differently and have their own advantages and limitations.

0.2 GPS

GPS uses a network of satellites orbiting the Earth to determine the location of an object. The GPS receiver on the object picks up signals from multiple satellites and uses the information to calculate its location. GPS is highly accurate and can provide location information with an error of just a few meters

0.3 IMU

On the other hand, an IMU uses sensors to measure the acceleration and rotation of an object. By integrating these measurements over time, it can determine the object's position and orientation. IMUs are commonly used in vehicles and drones, where GPS signals may not be reliable or available. However, IMUs are prone to accumulating errors over time, which can lead to significant inaccuracies in the distance calculations.

0.4 Python code to find distance using GPS data

```python
import math
fileName = input("Please enter the file name: ")
rawData = open(fileName, "r")

listOfLines = rawData.readlines()[1:]
coordinates = []

for line in listOfLines:
    data = line.split(",")
    coordinates.append([float(data[2]), float(data[3])])

totalDistance = 0


for i in range(1, len(coordinates)):
    x1,y1 = coordinates[i-1]
    x2,y2 = coordinates[i-0]

    distanceX = (x2-x1)*111045
    distanceY = (y2-y1)*87870.18

    distanceT = math.sqrt(distanceX*2 + distanceY*2)
    totalDistance += distanceT


#Printing Required Data
```

```
print("Total Distance covered(in m) = ", totalDistance)
```

0.5 Python code to find path using GPS data

```
import turtle
import math

#Opening The File
fileName = input("Please enter the file name: ")
rawData = open(fileName, "r")

#Choosing The Decimation Level
stepSize = int(input("Decimate The Input By Factor of : "))

#Reading The Contents
listOfLines = rawData.readlines()[1:]
coordinates = []

#Collecting Only Position Data
for line in listOfLines:
    data = line.split(",")
    coordinates.append([float(data[2]), float(data[3])])

#Turtle Screen
pathScreen = turtle.Screen()
pathScreen.title("Path Trace")

#Intializing Turtle
path = turtle.Turtle()
path.hideturtle()

#Drawing The Path
for i in range(stepSize+1,len(coordinates),stepSize):
        #Co ordintates of Tail & Head
        x1,y1 = coordinates[i-stepSize]
        x2,y2 = coordinates[i-0]

        #Calculating Individual Components
        distanceX = (x2-x1)*111045
        distanceY = (y2-y1)*87870.18

        #Final Vector Length & Angle
        totalDistance = math.sqrt( distanceX*2 + distanceY*2)
        angleCovered  = math.atan2(distanceY,distanceX)*(180/math.pi)

        #Adjusting The Angle
        if angleCovered >= 0:
                path.setheading(angleCovered)
        else:
                path.setheading(360+angleCovered)

```

```
        #Drawing The Line
        screeHeight = turtle.screensize()[1]
        path.forward((totalDistance*screeHeight/(1.78*60*10))*3)

turtle.done()
```

0.6 Path travelled

0.7 Python code for finding distance using sensors

```
import pandas as pd
import numpy as np
import sympy as sp
from scipy.integrate import cumtrapz



# Load the accelerometer, gyroscope and magnetometer data
accel_data = pd.read_excel('accelerometer_data.xlsx')
gyro_data = pd.read_excel('gyroscope_data.xlsx')
mag_data = pd.read_excel('magnetometer_data.xlsx')

Time = accel_data['time']
Time = Time.to_numpy()

x_axis = accel_data['ax']
```

```python
x_axis = x_axis.to_numpy()*9.8

y_axis = accel_data['ay']
y_axis = y_axis.to_numpy()*9.8

z_axis = accel_data['az']
z_axis = z_axis.to_numpy()*9.8


def cal(time, axis):
    stepsize =5
    axis = axis*stepsize
    prev_acc = 0.0
    prev_time = 0.0
    dis=0.0
    j=0
    for m,n in zip(time, axis):
        if(j!=0):
            # vel1 = abs(0.5*(m-prev_time)(n-prev_acc) + (m-prev_time)
            (min(n,prev_acc)))
            slope = (n-prev_acc)/(m-prev_time)
            vel = (((slope)(((m)2)/2)) + ((slope)(-prev_time)+prev_acc)(m))
            - (((slope)(((prev_time)*2)/2))
            + ((slope)(-prev_time)+prev_acc)*(prev_time))
            dis = dis + abs(vel*(m-prev_time))
        prev_time = m
        prev_acc = n
        j = j+1
    return dis

distance_arr = []
distance_arr.append(cal(Time,x_axis))
distance_arr.append(cal(Time,y_axis))
distance_arr.append(cal(Time,z_axis))
print(distance_arr)

distance = 0
for k in distance_arr:
    distance += k*k
print(f"Total distance travelled: {np.sqrt(distance)} meters")
```

0.8 Output

Using GPS data we got the distance as 714 meters. But using the sensors data(Accelerometer, Magnetometer and gyrocope) we got the distance as 910 meters. This is because of the noise in the data.

0.9 GPS Vs IMU

Therefore, in terms of distance calculation, GPS is considered to be more accurate than an IMU. However, GPS may not always be available or reliable, especially in indoor or urban environments where signals can be obstructed or weakened. In such cases, an IMU can provide useful information, although its accuracy may be limited over longer distances.