*※보고서 제목 양식 예 : 이름\_20240403*

|  |
| --- |
| **2024 ALTIS SW 개인 활동 보고서** |

|  |  |
| --- | --- |
| **활동 개요** | 시험발사 데이터 후처리 |
| **일시** | 2024년 8월 4일 |
| **작성자** | 김승창,노은지 |
| **활동 내용** | 1. 로켓의 이동 경로를 예측하기 위해서 에비오닉스가 측정한 가속도 값에 시간을 두 번 곱하여 X,Y,Z 축의 위치를 구하고 각 축 거리를 합친 최종 위치를 구했지만 예상한 그래프와 너무 다르게 나왔다.  2. 좀 더 정확하게 위치를 계산해보기 위해서 코랩을 통해 가속도 값에 시간을 두 번 적분했다. 그래도 예상과 다르게 나왔다.  3. 초기속도: 발사대 약 2.4m 를 통과하는 속도  데이터에서 2.32m를 통과하는데 0.335477s 가 걸려 6.9155m/s 보다 조금 빨랐을 것으로 예상한다.  최대고도는 396.21m이고 10.129682s 초에 도달했다.  전체 비행시간은 가속도의 값이 변하지 않는 구간을 찾아 대략 59초 걸렸다. |
| **활동 사진** | X축 위치  Y축 위치  Z축 위치  import pandas as pd  import numpy as np  import matplotlib.pyplot as plt  # 엑셀 파일 읽기  file\_path = '/content/data.xlsx'  sheet\_name = 'Avionics-Accel'  # 데이터 프레임 생성  df = pd.read\_excel(file\_path, sheet\_name=sheet\_name)  # 가속도 데이터 추출  time = df['time(sec)'] # 가정: 'Time' 열이 존재  acceleration\_x = df['Ax']  acceleration\_y = df['Ay']  acceleration\_z = df['Az']  dt = time.diff().mean() # 시간 간격 계산  def integrate\_acceleration(acceleration, dt, initial\_velocity=0, initial\_position=0):  velocities = np.zeros\_like(acceleration)  velocities[0] = initial\_velocity  for i in range(1, len(acceleration)):  velocities[i] = velocities[i-1] + acceleration[i-1] \* dt  positions = np.zeros\_like(acceleration)  positions[0] = initial\_position  for i in range(1, len(velocities)):  positions[i] = positions[i-1] + velocities[i-1] \* dt  return velocities, positions  # 각 축에 대해 적분 수행  velocities\_x, positions\_x = integrate\_acceleration(acceleration\_x, dt)  velocities\_y, positions\_y = integrate\_acceleration(acceleration\_y, dt)  velocities\_z, positions\_z = integrate\_acceleration(acceleration\_z, dt)  # 그래프 그리기  plt.figure(figsize=(12, 18))  # 가속도 그래프  plt.subplot(3, 1, 1)  plt.plot(time, acceleration\_x, label='Ax', color='r')  plt.plot(time, acceleration\_y, label='Ay', color='g')  plt.plot(time, acceleration\_z, label='Az', color='b')  plt.xlabel('Time (s)')  plt.ylabel('Acceleration (m/s^2)')  plt.title('Acceleration vs. Time')  plt.legend()  plt.grid(True)  # 속도 그래프  plt.subplot(3, 1, 2)  plt.plot(time, velocities\_x, label='Vx', color='r')  plt.plot(time, velocities\_y, label='Vy', color='g')  plt.plot(time, velocities\_z, label='Vz', color='b')  plt.xlabel('Time (s)')  plt.ylabel('Velocity (m/s)')  plt.title('Velocity vs. Time')  plt.legend()  plt.grid(True)  # 위치 그래프  plt.subplot(3, 1, 3)  plt.plot(time, positions\_x, label='Px', color='r')  plt.plot(time, positions\_y, label='Py', color='g')  plt.plot(time, positions\_z, label='Pz', color='b')  plt.xlabel('Time (s)')  plt.ylabel('Position (m)')  plt.title('Position vs. Time')  plt.legend()  plt.grid(True)  # 그래프 레이아웃 조정  plt.tight\_layout()  plt.show() |
| **활동 결과** |  |
| **계획** | 다른 방식으로 위치를 계산하고 각도도 포함해 비행 경로를 특정한다. |

2024년 8월 4일

작성자 : 김승창 (인)