3D BIRAKETAK

Aurreko praktikan egin den transbordadorea kokatu behar da espazioan. Bi biraketa desberdin aplikatu behar dira, bat Rodrigues formularekin eta beste bat kuaternoiekin. Praktika hau egin ahal izateko hainbat funtzio daude aurretik definituta biraketa moduluan:

- **rodriges**: Rodrigues-en formularekin biraketa matrizea egiteko.
- **quaternion**: Koaternoia sortzeko.
- **quat_dot_quat**: Bi kaoternoi biderkatzeko.
- **quat_dot_curve**: Koaternoia eta 3d kurba biderkatzeko.
- SLInterp: Interpolazioa egiteko.

Beste aldetik, diskretizatu deitzen den funtzio bat sortu dut eta denbora n alditan diskretizatzeko erabiltzen da. Biraketa modulua laugarren praktikarako ere erabiltzen denez, dena artxibo batean eginda dago (biraketak3d.py) eta biraketa modulua inportatu egin dut bi fitxategietan.

Rodrigues-ekin egindako programa

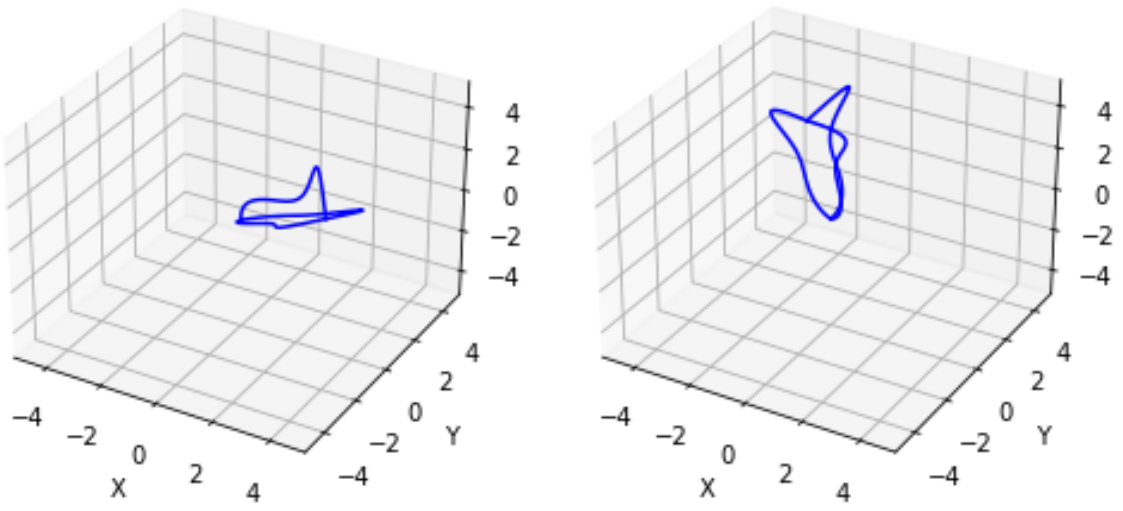
Lehenik, azken praktikaren transbordadorearen puntuekin bi irudi sortu ditut, bakoitza angelu eta norabide batekin. Lehenengoa `theta=pi/4` eta `w1 = [-1, -1, 0]`-ko norabidea izango du eta bigarrena `theta2 = pi/2` eta `w2 = [0, 1, 1]`. Irudia ondo marrazteko, spline bakoitza Rodrigues formularekin biraketa matrizea egin behar da. Hau egiteko, lehen definituta aurkitu dezakegun funtzioa erabili behar da.

```
puntuak_Rodr1 = np.dot(bir.rodrigues(theta,w1),puntuak1)
puntuak2_Rodr1 = np.dot(bir.rodrigues(theta,w1),puntuak2)
puntuak3_Rodr1 = np.dot(bir.rodrigues(theta,w1),puntuak3)
```

Hau egin eta gero, aurreko praktikaren berdina egin behar da transbordadorea ikusi ahal izateko. Bi aldiz egin beharko da, bat irudi bakoitzeko.

```
basis_splinea1_Rodr1, t_puntuak1_Rodr1 = splprep(puntuak_Rodr1, k=3, s=0) #Spline kubikoa, leuntasuna s=0
eta s=1 hartu
basis_splinea2_Rodr1, t_puntuak2_Rodr1 = splprep(puntuak2_Rodr1, k=3, s=0)
basis_splinea3_Rodr1, t_puntuak3_Rodr1 = splprep(puntuak3_Rodr1, k=3, s=0)

bs_kurba1_Rodr1 = splev(t_kurba, basis_splinea1_Rodr1)
bs_kurba2_Rodr1 = splev(t_kurba, basis_splinea2_Rodr1)
bs_kurba3_Rodr1 = splev(t_kurba, basis_splinea3_Rodr1)

fig = plt.figure()
ax = fig.add_subplot(projection='3d')

ax.plot(bs_kurba1_Rodr1[0], bs_kurba1_Rodr1[1], bs_kurba1_Rodr1[2], 'b-')
ax.plot(bs_kurba2_Rodr1[0], bs_kurba2_Rodr1[1], bs_kurba2_Rodr1[2], 'b-')
ax.plot(bs_kurba3_Rodr1[0], bs_kurba3_Rodr1[1], bs_kurba3_Rodr1[2], 'b-')
```

```
ax.set_xlim(-5, 5)
ax.set_ylim(-5, 5)
ax.set_zlim(-5, 5)
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')

plt.show()
```



Kuaternoiekin egindako programa

Azken pausuan kuaternoiak erabili behar dira matrizeen bidez biraketak definitzeko eta objektuaren bi orientazioren arten interpolatzeko, mugimenduaren animazioa sortuz. Horretarako biraketa moduluan dauden hainbat funtzio erabiliko ditugu. Lehenik, kuaternoien definizioa, p eta q-rekin:

```
p = bir.quaternion(theta, w1)
q = bir.quaternion(theta2, w2)
```

Orain, t diskretizatzen da bere funtzioarekin. Nik 50 alditan egitea erabaki dut geroago egingo dudan animazioa ondo ikusteko. Diskretizazio bakoitzerako SLInterp funtzioarekin p eta q kuaternoiekin eta s listan gordeko da bakoitza. Gero, begizta batean, irudi bakoitza sortzen da. Kode hau, ia aurrekoaren berdina da, baina untuak quat_dot_curve funtzioarekin kalkulatzen dira koaternoia eta puntua biderkatuz.

```
p1quater = bir.quat_dot_curve(s[i], puntuak1)
p2quater = bir.quat_dot_curve(s[i], puntuak2)
p3quater = bir.quat_dot_curve(s[i], puntuak3)
```

Azkenik, glob funtzioarekin irudiak hartzen dira, gero, irudiak izenez ordenatzen dira animazioa ondo ikusteko. Baita alderantziz ordenatzen dira irudiak animazioa aurrera eta atzera joateko. 25 fps jarri ditut irudi fluidoak ikusteko. Bukatzeko, animazioa sortzen da ImageSequenceClip eta write_gif-rekin. Animazioa gif bezala gordetzen da karpetan.

```
#Irudiak karpetatik atera
file_list = glob.glob('./figures/*.png')

fps = 25
#Irudiak izenez alfabetikoki ordenatu
seq = natsort.natsorted(file_list)
#Irudiak izenez alfabetikoki aldreves ordenatu
seq_ald = natsort.natsorted(file_list, reverse = True)
#Sortu sekuentzia
clip = mpy.ImageSequenceClip(seq+seq_ald, fps=fps)

#Gorde gifa
clip.write_gif('animazioa.gif')
```