KLASSIK MEXANIKANING MURAKKAB HODISALARINI PYTHON ANIMATSIYALARI ORQALI VIZUALLASHTIRISH

Usmonova Mohlaroy
Andijon davlat pedagogika instituti
Aniq va Tabiiy fanlar kafedrasi o’qituvchisi
Zokirova Mohlaroy
Fizika va astronomiya yo’nalishi 3-bosqich talabasi
https://doi.org/10.5281/zenodo.20699602

Annotatsiya

Maqola fizik hodisalarni o'qishni soddalashtirishda Python animatsiyalarining roli haqida. Klassik mexanikaning ikkita asosiy masalasi (mayatnik va snaryad harakati) SciPy, NumPy va Matplotlib kutubxonalari orqali vizuallashtirilib, har bir animatsiya kodining fizik va matematikasini tushuntirildi. Runge-Kutta va Eyler usullari amaliy misollar bilan ko'rsatildi. Maqola fizika o'qituvchilari va Python programmalarini o'rganayotgan talabalar uchun foydalidir.

Kalit so'zlar: Python animatsiya, vizualizatsiya, differensial tenglamalar, klassik mexanika, ta'lim texnologiyasi.

Fizika — bu tabiat qonuniyatlarini o'rganadigan fan bo'lsa, dasturlash — bu qonuniyatlarni raqamli dunyoda jonlantirish vositasidir. Python o'zining sodda sintaksisi va boy kutubxonalari bilan murakkab fizik jarayonlarni vizuallashtirish uchun eng qulay tildir. Fizikadagi murakkab hodisalarni animatsiyalar orqali tushuntirish ancha samarali va qiziqarli qiladi. Bu

animatsiyalarda Pythondan foydalanish esa har bir xususiy holat uchun yangi animatsiyalarni oson yaratishga yordam beradi.Ushbu maqolada biz **Mayatnik harakati, Snaryad harakati** Python yordamida animatsiya qilamiz.

Barcha animatsiyalar uchun biz asosan uchta kutubxonadan foydalanamiz:

NumPy — Matematik hisob-kitoblar uchun

Matplotlib — Grafik va animatsiya uchun

SciPy — Differensial tenglamalarni yechish uchun Birinchi navbatda mayatnik harakatini ko’rib chiqsak:

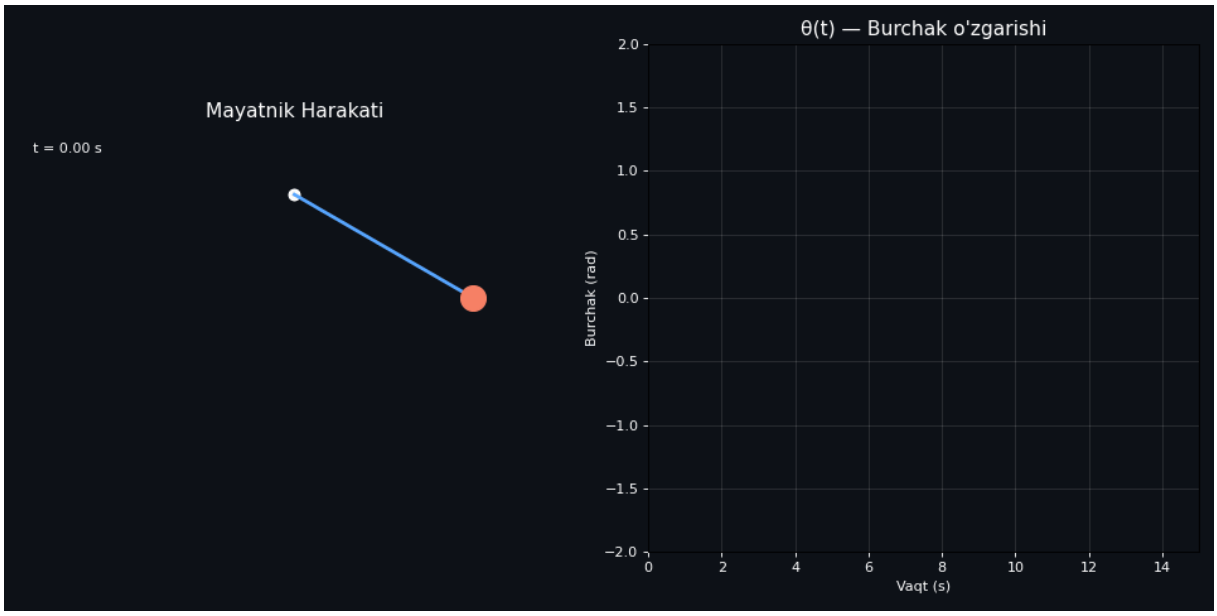
pendulum_ode funksiyasi — Bu funksiya ikkinchi tartibli differensial tenglamani birinchi tartibli sistema ko'rinishiga o'tkazadi. Matematikada bu standart usul: bir o'zgaruvchi (theta) o'rniga ikkitasini (theta va omega) kuzatamiz.

solve_ivp — SciPy kutubxonasidan olingan bu funksiya differensial tenglamalarni Runge-Kutta usulida yechadi. rtol=1e-8 parametri hisoblash aniqligini oshiradi — fizik tajribalarda aniqlik muhim.

FuncAnimation — Matplotlib-ning animatsiya mexanizmi. Har bir frame uchun animate funksiyasi chaqiriladi va grafik elementlari yangilanadi. blit=True faqat o'zgargan qismlarni qayta chizadi — bu tezlikni oshiradi.

```python
# maqola.py ga ...
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.animation as animation
from scipy.integrate import solve_ivp

# ============================================
# 1-QISM: MAYATNIK
# ============================================
g = 9.81
L = 1.5
theta0 = np.pi / 3
omega0 = 0.0

def pendulum_ode(t, y):
    theta, omega = y
    return [omega, -(g / L) * np.sin(theta)]

t_eval = np.linspace(0, 15, 1500)
solution = solve_ivp(pendulum_ode, (0, 15), [theta0, omega0],
                     t_eval=t_eval, method='RK45', rtol=1e-8)
theta = solution.y[0]
x_bob = L * np.sin(theta)
y_bob = -L * np.cos(theta)

fig1, (ax1, ax2) = plt.subplots(1, 2, figsize=(12, 6))
fig1.patch.set_facecolor('#0d1117')

ax1.set_xlim(-2, 2); ax1.set_ylim(-2, 0.5)
ax1.set_aspect('equal'); ax1.set_facecolor('#0d1117')
ax1.set_title("Mayatnik Harakati", color='white', fontsize=14)
ax1.axis('off')

ax2.set_xlim(0, 15); ax2.set_ylim(-2, 2)
ax2.set_facecolor('#0d1117')
ax2.set_title("θ(t) — Burchak o'zgarishi", color='white', fontsize=14)
ax2.set_xlabel("Vaqt (s)", color='white')
ax2.set_ylabel("Burchak (rad)", color='white')
ax2.tick_params(colors='white')
ax2.grid(True, alpha=0.2)

ax1.plot(0, 0, 'o', color='white', markersize=8)
rod_line, = ax1.plot([], [], '-', color='#58a6ff', linewidth=2.5)
bob_circle, = ax1.plot([], [], 'o', color='#f78166', markersize=18)
trace_line, = ax1.plot([], [], '-', color='#3fb950', alpha=0.3, linewidth=1)
time_text = ax1.text(-1.9, 0.3, '', color='white', fontsize=10)
graph_line, = ax2.plot([], [], color='#f78166', linewidth=1.5)

trace_x, trace_y = [], []

def animate1(frame):
    i = min(frame * 3, len(x_bob) - 1)
    rod_line.set_data([0, x_bob[i]], [0, y_bob[i]])
    bob_circle.set_data([x_bob[i]], [y_bob[i]])
    trace_x.append(x_bob[i]); trace_y.append(y_bob[i])
    if len(trace_x) > 80:
        trace_x.pop(0); trace_y.pop(0)
    trace_line.set_data(trace_x, trace_y)
    time_text.set_text(f't = {t_eval[i]:.2f} s')
    graph_line.set_data(t_eval[:i], theta[:i])
    return rod_line, bob_circle, trace_line, time_text, graph_line

anim1 = animation.FuncAnimation(fig1, animate1, frames=500, interval=30, blit=True)
plt.tight_layout()
anim1.save('mayatnik_animatsiya.gif', writer='pillow', fps=30, dpi=80)
print("✅ mayatnik_animatsiya.gif saqlandi")
plt.close()
```

Bu animatsiya vaqt o’tishi bilan mayatnik parametrlari o’zgarishini mayatnik harakati va diagramma orqali tushuntirilgan. Bu animatsiya orqali jarayonni tushuntirish formulalarni real ko’z bilan ko’rishga yordam beradi.



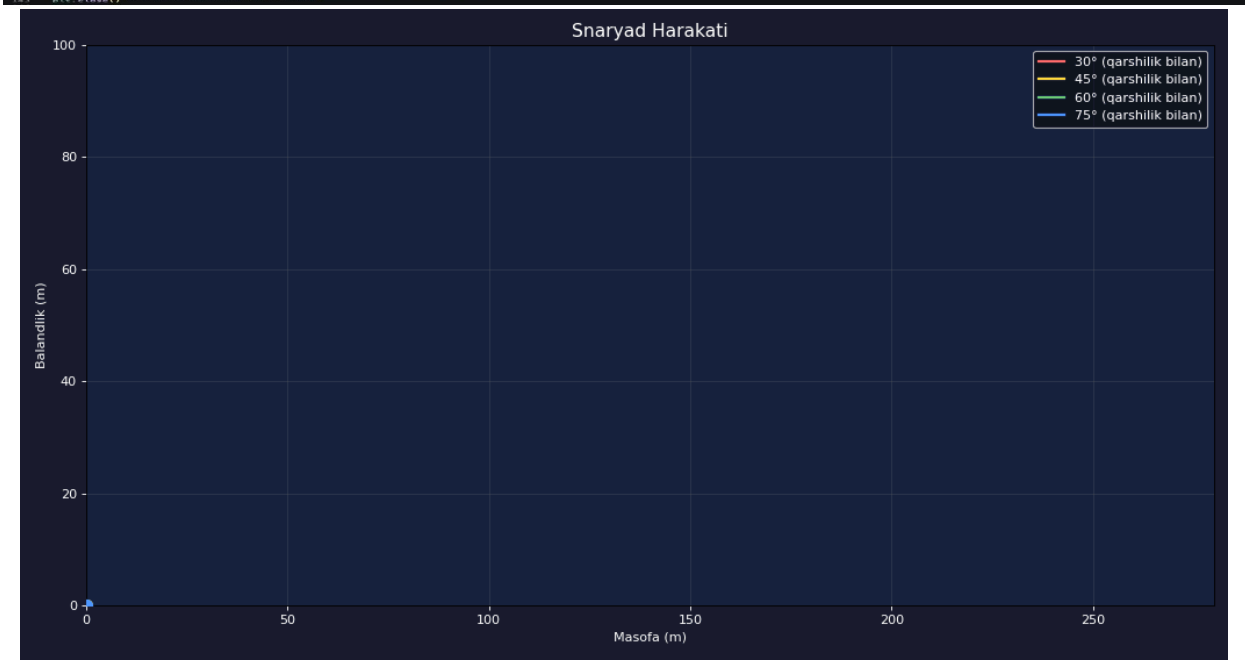
Snaryad harakati

Snaryad (proektil) harakati — ikki o'lchovli kinematikaning klassik misoli. Bu harakatlarni havoning qarshilik kuchi bor va havo qarshilik kuchi mavjud bo’lmagan holatini ko’rib chiqamiz.

Eyler usuli — compute_trajectory funksiyasida biz sodda lekin samarali Eyler integratsiyasidan foydalanamiz. Har bir dt qadamda tezlik va pozitsiyani yangilaymiz. dt = 0.01 s — bu hisoblash va aniqlik o'rtasidagi muvozanat.

Koeffitsient k = 0.02 — mis to'p uchun taxminiy qiymat.

Ikki chiziq — Har bir burchak uchun ikkita traektoriya ko'rsatiladi: real (havo qarshiligi bilan) va ideal. Bu fizikaning muhim tushunchasini vizual ko'rsatadi: havo qarshiligi nafaqat masofani kamaytiradi, balki optimal burchakni ham 45°dan past tomonga siljitadi.



Ushbu maqolada biz Python dasturlash tilining **NumPy**, **Matplotlib** va **SciPy** kutubxonalaridan foydalanib, klassik fizikaning ikkita muhim hodisasini animatsiyalar orqali vizuallashtirdik. Python orqali fizikani animatsiya qilish — nafaqat ta'limning samarali vositasi, balki hozirgi yosh fizikani murakkab jarayonlarni tezda modellash va tajriba qilish imkonini beradi.

Adabiyotlar, References, Литературы:

1.Goldstein, H., Poole, C. & Safko, J. (2002). Classical Mechanics (3rd edition). Addison-Wesley Publishing Company. ISBN: 978-0201657029. Shuning bilan: Klassik mexanikaning eng kuchli akademik manba. Mayatnik, snaryad harakati, Lagrange va Hamilton mexanikasi.

2. Marion, J. B. & Thornton, S. T. (2004). Classical Dynamics of Particles and Systems (5th edition). Brooks/Cole Publishing. ISBN: 978-0534408961. Shuning bilan: Mexanika masalalarini amaliy misollar bilan tushuntiradi.

3. Harris, C. R., Millman, K. J., van der Walt, S. J., et al. (2020). "Array programming with NumPy". Nature, 585, 357-362. DOI: https://doi.org/10.1038/s41586-020-2649-2 Shuning bilan: NumPy asosiy texnika va vektorlashtirilgan operatsiyalar.

4. NumPy Documentation. (2024). https://numpy.org/doc/stable/ Shuning bilan: Rasmiy dokumentatsiya, API reference.
5. Matplotlib Documentation. (2024). https://matplotlib.org/stable/contents.html Shuning bilan: Grafikalar, animatsiya, tikzplotlib.