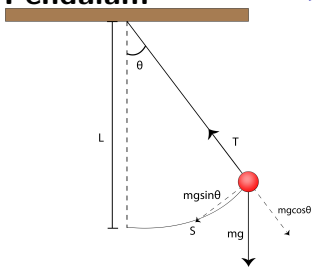
Demonstrations

Nuestro objetivo principal es motivar a los estudiantes sobre el estudio de la computación como herramienta para la exploración de diversidad de problemas junto con su resolución, usando Python como herramienta principal. Esto con el fin que en el futuro puedan desarrollar un razonamiento lógico aplicado a la creación de algoritmos.

Se busca que con esta idea se puedan implementar estos ejemplos como experimentos demostrativos, análogos a los que actualmente se realizan en Física 1 y Física 2.

`https://github.com/ComputoCienciasUniandes/Demonstrations`

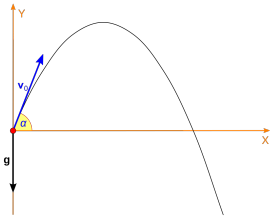
- **Pendulum**



$$\ddot{\theta} = -\frac{g}{L}\sin\theta$$

- Fuerzas
- Condiciones iniciales interactivas

- **ProjectileMotion**

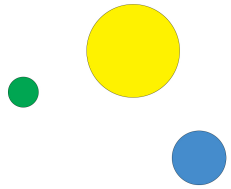


$$\ddot{x} = -\frac{\beta}{m}\dot{x}$$

$$\ddot{y} = -\frac{\beta}{m}\dot{y} - mg$$

- Fuerzas
- Condiciones iniciales interactivas

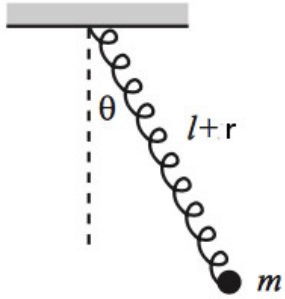
- **SolarSystem**



$$\ddot{\vec{r}}_i = G\sum_{i\neq j}^{N}\frac{m_j}{|\vec{r}_{ij}|^3}(\vec{r}_j - \vec{r}_i)$$

- Fuerzas
- Datos reales de NASA Horizon

- **SpringPendulum**

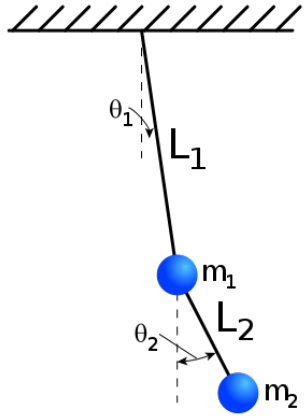


$$\ddot{r} = (l+r)\dot{\theta}^2 + g\cos\theta - \omega^2 r$$

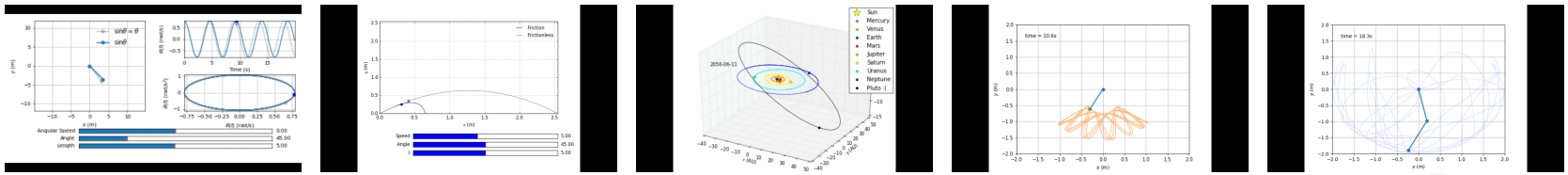
$$\ddot{\theta} = -\frac{1}{l+r}\left(2\dot{r}\dot{\theta} + g\sin\theta\right)$$

- Energías (Lagrangiano)

- **DoublePendulum**

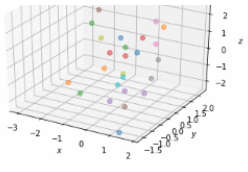


- Energías (Lagrangiano)
- Uso de cálculo simbólico



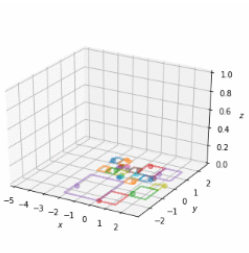
Demonstrations

- **astrohut**
  (`https://jsbarbosa.github.io/astrohut/`)
  astrohut is a NBody gravity simulator that aims to help students understand many body systems, as well as motivating the use of computational tools to solve physical problems. Written in Python, with the core functions in C.

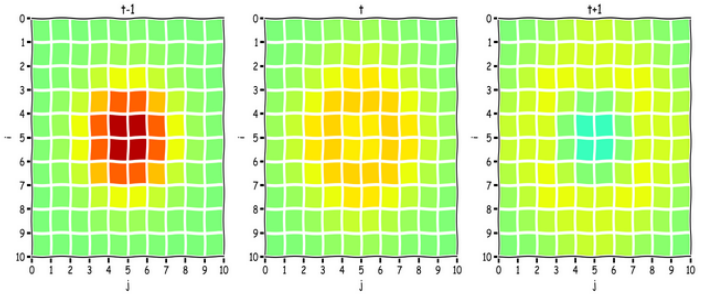


```python
import numpy as np
import astrohut as ah
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

G = 1.0
m = 1.0

pos = np.random.normal(size=(25, 3))
speeds = ah.generateSpeeds(pos, G, m)

system = ah.createArray(pos, speeds)

sim = ah.Simulation(system, dim = 3, dt = 1e-3, G = G, mass_unit = m)

sim.start(1000, save_to_array_every = 25)

# if boxes are wanted: boxed = True, else: boxed = False
ani = sim.makeAnimation(boxed = True)

# ani.save("random3d.gif", writer="imagemagick", dpi = 72, fps = 12)
plt.show()
```

disk3d



- **rippleTank**
  (`https://jsbarbosa.github.io/rippleTank/`)
  rippleTank is a simulator that aims to help students understand how waves behave on a ripple tank, as well as motivating the use of computational tools to solve physical problems. Fully written with Python.



Documentation

rippleTank is divided in four scripts: *masks.py*, *sources.py*, *tank.py*, *examples.py*. They contain the definition of three classes: *Mask*, *Source* and *rippleTank*.

Root class is *rippleTank* which defines the whole space in which the simulation will take place. Perturbations are included with *Source* objects and obstacules with *Mask* objects.

- Documentation
  - masks
  - sources
  - tank

Examples