Final Project:

Simulating Balance Recovery and Fluid Interaction in a 2D Human-Like Body

Gerardo L. Morales Torres

CMSE 802

April 21, 2025

GitHub Repository: https://github.com/moral201/cmse802_project.git

1. Background and Motivation

This project presents a modular, physics-based simulation and animation framework designed to explore how a simplified 2D human-like figure can maintain balance and react to external disturbances, such as falling fluid particles. The primary goal was to simulate a figure capable of staying upright through reactive stepping and arm control, using only basic physics and real-time joint torques. This naturally led to the central question: *Can we create an animation that appears physically plausible using simple control logic and mathematical modeling?*

Beyond technical implementation, this project was personally motivated by a long-standing interest in movies and video games. As someone fascinated by interactive media, I've often wondered how physics simulations work behind the scenes. This project provided a creative and practical opportunity to explore that curiosity—transforming abstract mathematical equations into a dynamic and visually engaging animation. It also allowed me to apply concepts from control theory and physics in a way that felt tangible and system-oriented rather than purely theoretical.

Technically, the approach draws on classical methods from robotics and control, particularly the inverted pendulum model [1], which is widely used to approximate human balance. While often discussed in research and textbooks, here it formed the foundation for simulating both the stepping behavior and overall body stabilization. The simulation expands on this base to incorporate articulated arms and fluid-particle collisions, producing a cohesive and testable environment for modeling balance and interaction.

2. Methodology

To explore how a simplified human-like body could maintain balance and respond to environmental disturbances, the project was approached as a modular simulation problem. The simulation was divided into several core subsystems: the reactive stepping controller, articulated arm control, and environmental interaction through falling fluid particles. Each subsystem was implemented using basic physics and control principles, with an emphasis on clarity, reusability, and real-time behavior.

Models Used

- **PD Controller and Reactive Stepping**
  Both the legs and the torso/upper body were modeled as inverted pendulums, capturing the dynamics of a human balancing over a narrow base. A proportional-derivative (PD) controller was applied at each joint to compute torques based on angular position and velocity [2]. This allowed the system to respond to imbalance by shifting the center of mass through leg motion and torque adjustments at the joints.

  To simulate a loss of balance, the simulation applies an intentional initial "push"—implemented as a horizontal displacement of the torso's center of mass. This causes the upper body to tip, requiring a corrective response to avoid falling. The system responds

using a reactive stepping strategy, where the legs dynamically reposition to extend the base of support. This realigns the center of mass over the feet, mimicking how humans instinctively step to regain balance after a disturbance.

Importantly, the torso is not given a fixed target angle. Instead, the PD controllers at the hip joints minimize the relative angle between the legs and torso, indirectly stabilizing the upper body. As the legs step and reposition, torque applied at the joints brings the torso back to an upright configuration.

The control torque for each joint is computed using:

$$\tau = -K_p\theta - K_d\omega$$

Where:

- τ is the applied torque
- θ is the current joint angle
- ω is the angular velocity
- K_p is the proportional controller gain
- K_d is the derivative controller gain

This controller provides smooth and responsive joint behavior and, when combined with dynamic stepping, allows the body to recover balance in a physically plausible way.

- **Arm and Legs Control**
  Each arm was modeled as a two-link chain connected to a fixed shoulder joint. PD controllers were applied independently at each joint to maintain a natural resting pose, while still allowing reactive motion during collisions with fluid particles. Although each joint computes its torque independently, the same gain values are used across all arm segments for simplicity.

  To demonstrate dynamic control, the right arm was programmed to perform a waving gesture by applying a sinusoidal variation to its target angle over time. Specifically, the target elbow angle was updated using a sine wave function:

$$\theta_{target} = \theta_{rest} + 0.3 \cdot \sin(2\pi \cdot 1.5t)$$

  This created a smooth, periodic “waving hi” motion synchronized with the simulation time.

  The angular position and velocity of each segment were updated using:

$$\alpha = \frac{d\omega}{dt} = \frac{\tau - 0.2\omega}{ml^2}$$

$$\omega_{new} = \omega + \alpha \cdot dt$$

$$\theta_{new} = \theta + \omega_{new} \cdot dt$$

Where:

- α is angular acceleration
- m is the mass of the segment
- l is the length of the segment
- dt is the time step

- **Fluid Particle System**
  Environmental disturbances were modeled using particles falling under gravity [3]. Their motion was updated using simple kinematic rules, and collisions with the body were resolved using vector reflection based on the surface normal. This approach provided a lightweight yet visually effective approximation of environmental interaction without affecting the body's dynamics.

$$y_v -= g \cdot dt$$
$$x_p += x_v \cdot dt$$
$$y_p += y_v \cdot dt$$

Where:

- (x_p, y_p) is particle position
- (x_v, y_v) is particle velocity
- g is gravity

- **Velocity Reflection Upon Collision**:

  When particles come into contact with the body, their velocities are reflected based on the surface normal at the point of impact. This produces visually realistic, directionally consistent bounces from arms, legs, and torso.

$$\overrightarrow{v_{new}} = \vec{v} - 1(1 + e)(\vec{v} \cdot \vec{n})\vec{n}$$

Where:

- $\vec{n}$ is surface normal
- e is elasticity (coefficient of restitution)

Although these particles do not physically affect the body, their consistent collisions reinforce the visual realism of the simulation and illustrate joint coordination under continuous environmental interaction.

All of these components were integrated in main_simulation.py, with reusable functions defined in physics_utils.py and constants organized in global_constants.py. The simulation was executed and visualized in a Jupyter notebook, allowing for animation, tuning, and testing in an interactive environment.

Validation Metrics

Since the nature of this project is animation-based, performance was primarily evaluated through visual inspection and behavioral correctness. The key qualitative metrics included:

- **Stability**: The figure should maintain an upright posture throughout the simulation without falling, even under continuous particle interaction.
- **Recovery**: When perturbed by the initial push, the stepping controller should visibly reposition the legs to regain balance.
- **Plausibility**: Arm motion and fluid-particle interactions should appear smooth, coherent, and physically reasonable.
- **Numerical Robustness**: The simulation should run without instability, jittering, or unintended clipping artifacts.

To verify these behaviors, multiple test runs were conducted using randomized particle inputs and varied initial conditions. In addition, unit tests were implemented to validate key physics functions such as torque computation and particle reflection logic, ensuring correctness and reliability at the functional level.

3. Results

The final simulation successfully produced a full-body animation demonstrating the intended behaviors: balance recovery through reactive stepping, articulated arm motion, and consistent interaction with falling fluid particles. The system maintained stability under repeated environmental collisions and responded naturally to the initial push disturbance.

Key visual results included:

- The body remained upright by dynamically shifting its leg position in response to tipping.
- Arms maintained a resting pose using PD control and visibly reacted to particle collisions.
- The right arm performed a waving motion, driven by a sinusoidal target angle.
- Particles fell continuously and bounced realistically off the figure without penetrating or sticking.
- No instability, jitter, or visual artifacts were observed across extended simulations.

The animation video can be found on the repository under the results folder, with the name: final_implementation.mp4. Below is a still frame from the animation that illustrates the final implementation in action:

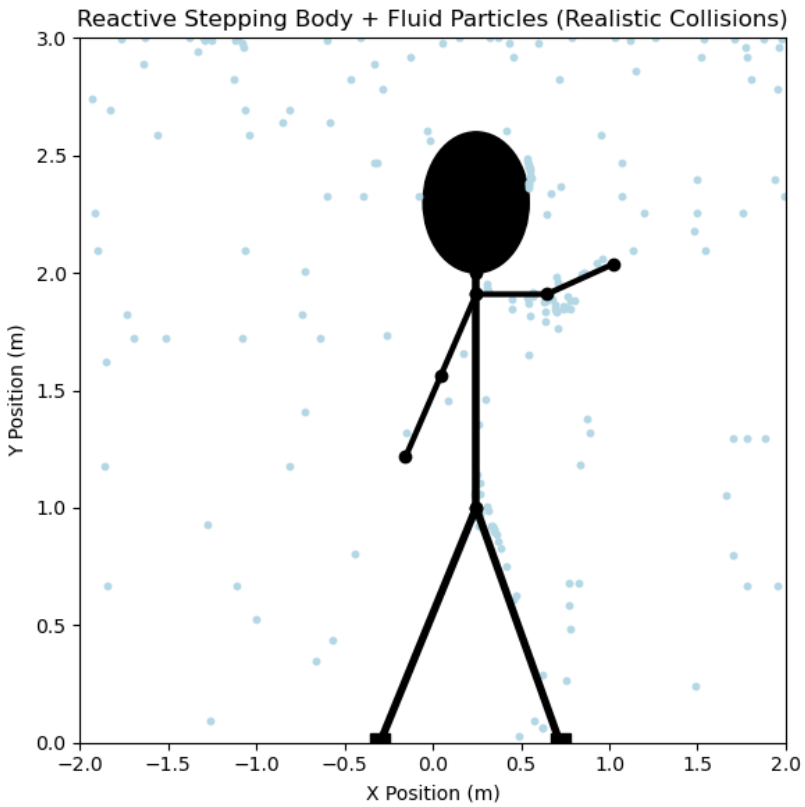

Figure 1. Animation Screenshot

All of these behaviors were generated in real time using only core physics and mathematical rules defined in the simulation code. The results visually confirm that the system functions as intended and meets the project's goals.

4. Synthesis and Discussion

This project provided valuable insight into how balance, joint control, and environmental interaction can be modeled using physics-based simulation. One of the key takeaways was that visually realistic behavior can emerge from relatively simple equations when components like PD control, rigid body dynamics, and collision mechanics are properly integrated.

The animation successfully demonstrated balance recovery using reactive stepping, articulated arm control, and visually coherent fluid-particle interaction. Even though the particles had no physical effect on the body, their collisions and reflections reinforced the perception of a responsive, dynamic system.

Obstacles Encountered

Several challenges arose during development:

- Balancing the torso without a direct target angle required understanding how to stabilize the system through leg torque and stepping rather than absolute angle constraints.

- Particle collisions were initially overly simplified and reflected only vertically. Resolving direction using surface normals improved realism.
- Unexpected visual behavior, such as particles sliding along the body, highlighted the limits of not modeling friction — but also added an unintentional yet plausible effect.

What I Would Do Differently

If given more time, focus would be given to:

- Introduce foot-ground contact constraints to improve stepping realism.
- Add joint limits to prevent hyperextension.
- Explore adaptive or learning-based control to adjust joint responses dynamically.
- Begin modeling inter-particle interactions for more complex fluid behavior.

Final Reflection

This project successfully implemented a modular 2D simulation of a human-like figure capable of maintaining balance through reactive stepping and arm control, even under continuous environmental disturbance. By combining biomechanical modeling, PD control, and simplified fluid dynamics, the simulation achieved a visually and physically coherent representation of dynamic human motion. The approach proved effective in meeting the project's goals. The use of separate modules for body dynamics, control logic, and fluid simulation made the system easy to test, extend, and debug. The reactive stepping controller allowed the system to recover from disturbances in a natural and repeatable way. While the project remains limited to 2D and simplified fluid models, it establishes a strong foundation for future work in biomechanical simulation. The experience also provided a better understanding of how game and animation physics systems are constructed and how rewarding it can be to bring mathematical models to life visually.

5. References

[1] "Control Tutorials for MATLAB and Simulink - Inverted Pendulum: PID Controller Design." Accessed: Feb. 09, 2025. [Online]. Available: https://ctms.engin.umich.edu/CTMS/index.php?example=InvertedPendulum&section=ControlPID

[2] Ogata, K. (2010). *Modern Control Engineering.* Prentice Hall. (For foundational PD control concepts)

[3] M. Müller, D. Charypar, and M. Gross, "Particle-Based Fluid Simulation for Interactive Applications," Proceedings of ACM SIGGRAPH, 2003. Accessed: Feb. 09, 2025. [Online]. Available: https://matthias-research.github.io/pages/publications/sca03.pdf