PRD - Player Movement System (Mine now Fighter)

Problem Statement

The current movement feels floaty and unresponsive. Players cannot control spacing, angle changes, or timing reliably. This reduces combat clarity and makes duels feel inconsistent. A fast-paced 1v1 fighter requires tight, predictable, grounded movement to support skill-based combat.

Goals

- Create a fast, responsive, camera-relative movement system.
- Support walk, run, strafe, and backward movement with clean state transitions.
- Ensure movement feels grounded and readable with boxing/karate stance principles.
- Integrate cleanly with combat (attacks pause movement, movement resumes after recovery).
- Build a foundation that supports future features (dodge, dash, advanced combat).

Movement Philosophy

Stance

Boxing/Karate hybrid stance: upright, high guard, light bounce at idle.

Footwork

Hybrid: light bounce at idle, grounded movement during walk/strafe/backward.

Movement Tempo

Fast footwork. Quick repositioning, strong spacing control.

Turning

Rotate smoothly but very fast (0.05–0.1s). No snapping.

Stop Behavior

Instant stop: velocity immediately resets to 0 when input is released.

Root Motion

All movement uses root-motion animations for realistic foot placement.

Feature Description

A 3D camera-relative movement system supporting walk, run, strafe (left/right), and backward movement.

Movement transitions must be fast and grounded, supporting a small 1v1 arena fighter.

Root motion ensures realistic foot placement.

Rotation follows input direction quickly, and camera follows character facing direction with minimal lag.

Movement must feel precise, clean, and built for competitive spacing-based combat.

Movement States

- Idle
- Walk (forward/side/backward)
- Run (forward only)
- Strafe Left / Strafe Right
- Stop (instant)

- Crouch (future)
- Block (future)

State Transition Rules

- Idle → Walk: input magnitude > threshold
- Walk → Run: input magnitude > high threshold
- Run → Idle: input released
- Walk ↔ Strafe ↔ Backward: based on input direction
- Any Movement → Stop: input released
- Movement is overridden during attack animations (future PRD)

Functional Requirements

Input & Direction

- Movement uses camera-relative input.
- Direction recalculates every frame.
- Diagonal input normalized (no speed gain).
- Deadzone threshold applied.

Rotation

- Character rotates smoothly to new direction within 0.05–0.1s.
- Rotation updates even before animation fully blends.
- Camera follows rotation within 2–3 frames.

Walk

- Root-motion walk animation blended in cleanly.
- Lateral/backward walk speed slower than forward walk.

Run

- Forward-only run state.
- Activated by strong input.
- Must feel fast but grounded.
- Cannot attack during run (must return to walk/idle first).

Instant Stop

- Velocity immediately set to 0.
- Animation blends to idle within 0.02–0.05s.
- No sliding allowed.

Animation Requirements

- Root-motion used for all movement.
- Blend tree supports 360° transitions (forward, backward, left, right).
- No turn-in-place animations; turning is procedural + blended.
- Idle bounce transitions smoothly into grounded movement.
- Foot placement must look stable during fast pivots.

Camera Requirements

- Camera always aligns with player facing direction.
- No lag; minimal smoothing to avoid jitter.
- Camera adjusts smoothly during run/fast turns.
- Must maintain readability of opponent in small arena.

Non-Functional Requirements

Responsiveness

- Movement delay < 80ms
- Rotation alignment < 0.1s
- Camera follow < 2–3 frames

Performance

- Movement logic < 1ms
- Animation blending < 2ms

Stability

- No drift, sliding, or floatiness.
- Works on uneven terrain.

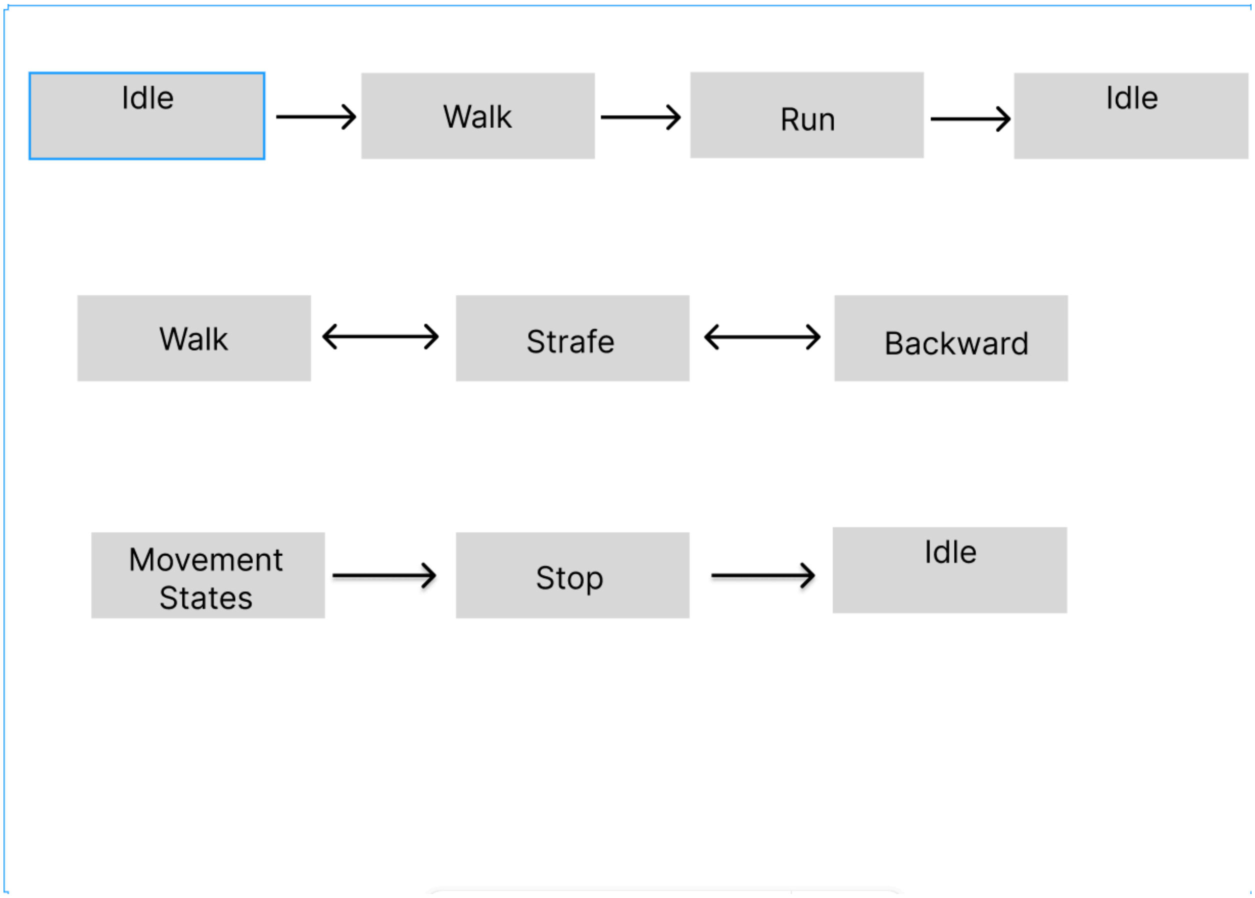
Acceptance Criteria

- Player movement feels fast, grounded, and predictable.
- Walk → Run → Stop transitions feel clean and immediate.
- No foot sliding or animation popping.
- Camera remains stable during all movement.
- Playtesters report “movement feels responsive and controlled.”

Risks & Constraints

- Root-motion requires high-quality animations.
- Fast movement may challenge hitbox alignment later.
- Camera must avoid jitter during rapid pivots.
- Run must not overpower spacing (must be tuned carefully).
- Any future dodge/dash must not break current movement flow.

Movement System Flow V1



Insights

Problem that movement system solves

It solves the issue of unresponsive and floaty movement, which makes spacing, timing, and positioning inconsistent in combat. Players can't read their own distance or the opponent's. This reduces fairness, skill expression, and overall combat clarity.

How does the movement system effect combat?

Precise movement improves combat by giving players reliable control over distance, strafing angles, and timing windows. This creates more meaningful decisions when to engage, disengage, or bait which directly raises skill ceiling and combat satisfaction.

What metrics will measure the good movement?

Good movement can be measured using:

- Input-to-movement response time (<80ms target)
- Hit accuracy during motion (higher accuracy = better control)
- Player-reported control clarity (post-match survey)
- Reduced unintended collisions or drift

Learnings

Working on the movement system showed me how much depth goes into features that seem simple on the surface.
I learned that every movement detail speed, rotation, transitions, stop behavior directly affects player decision-making and combat clarity.
Translating this into requirements forced me to think like a Product Owner: identifying problems, defining goals, and breaking a complex feature into clear, testable parts.

I also saw how technical constraints shape design choices, like using root motion and blend trees to make footwork smooth and grounded.
Overall, I learned how design, animation, and player psychology come together, and how important it is to communicate decisions clearly.