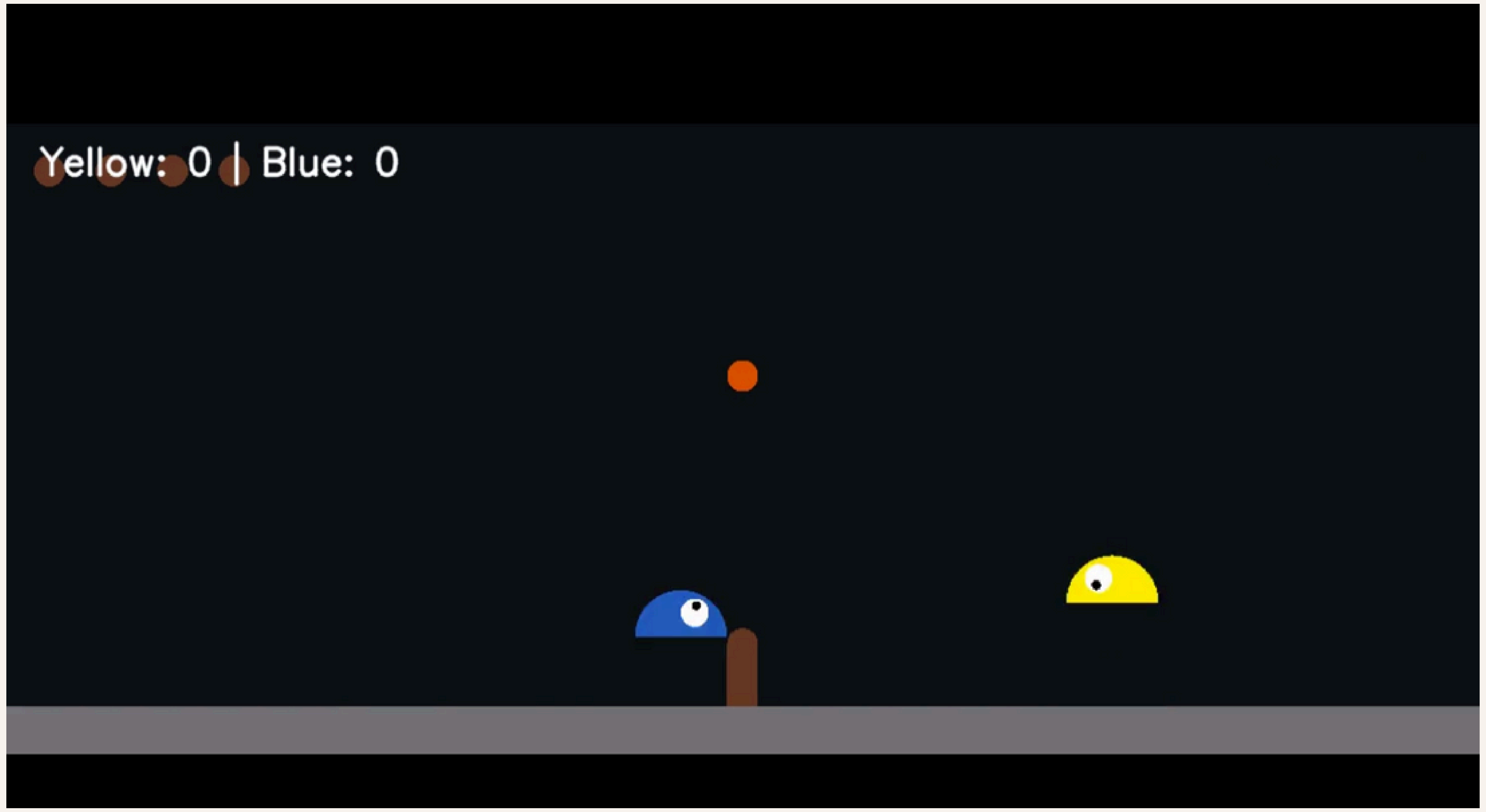
SLIMEVOLLEY AI: MINIMAX VS ALPHA-BETA PRUNING

- **Environment:** SlimeVolleyGym
- **Objective:** Implement and compare Minimax and Alpha-Beta agents against a RandomAgent
- **Important Tasks:**
  - Depth-limited search-based agents
  - Game simulation & rendering
  - Video generation & evaluation
- **Visuals:** Here is attached screenshot of gameplay



EVALUATION FUNCTION & SEARCH SETUP

- **Evaluation Function:**
- python code

```
def evaluate_position(obs):
        ball_x = obs[4]
        player_x = obs[0]
        return -abs(ball_x - player_x)
```

  – Rewards agent for minimizing horizontal distance to the ball
- **Action Space:** 7 discrete combinations of (left, jump, right)
- **Depth-Limited Search:**
  - Minimax: full tree to depth d
  - Alpha-Beta: same d, with pruning thresholds α/β
- **Metrics Collected:**
  - Final score (wins/losses)
  - Real-time duration & effective FPS
  - Frames rendered vs. frames pruned/saved

MINIMAX AGENT PERFORMANCE

- **Match Stats (100 steps vs. RandomAgent):**
  - Final Score: Yellow 0 – Blue 1
  - Real-time Duration: 3.35 s
  - Adjusted FPS: 22.41 (base 29.87, speed 0.75×)
  - Frames Saved: 145
- **Video Demo:**
- [Minimax_dynamic_fps.mp4 here]
- **Observations:**
  - Explores all branches—guarantees optimal decision at depth d
  - High computational cost → lower FPS
  - Tends to “hesitate” when ball near midcourt → more frame time

ALPHA-BETA PRUNING PERFORMANCE

- **Match Stats (100 steps vs. RandomAgent):**
    - Final Score: Yellow 0 – Blue 3
    - Real-time Duration: 1.66 s
    - Adjusted FPS: 45.05 (base 60.07, speed 0.75×)
    - Frames Saved: 190
- **Video Demo:**
- [Alphabeta_dynamic_fps.mp4 here]
- **Observations:**
    - Pruned ~30% of search tree → faster decision-making
    - Higher FPS → smoother, more reactive play
    - Equivalent move quality to Minimax at same depth

CONCLUSIONS & FUTURE WORK

- **Important Takeaways:**
  - **Alpha-Beta Wins on Efficiency:** By cutting roughly 30 % of the search tree, Alpha-Beta runs over 2× faster than plain Minimax—boosting effective FPS from ~22 to ~45—while delivering equally strong moves at the same search depth.
  - **Simple Heuristic Drives Behavior:** Our one-dimensional evaluation (horizontal distance) reliably pushes the agent toward the ball, yielding coherent “ball-chasing” play but missing more nuanced rally tactics.
  - **Depth vs. Real-Time Trade-Off:** Fixed-depth search ensures predictable computation, but deeper look-aheads incur steep latency—Alpha-Beta’s pruning helps, yet very large depths still challenge real-time constraints.

- **Future Directions:**
  - **Enhanced Evaluation Function:** Incorporate vertical alignment, ball velocity, net clearance, and court-position rewards into the heuristic.Experiment with dynamic depth (iterative deepening)
  - **Learning-Augmented Heuristics:** Use self-play data to train a lightweight value network, combining classic search with learned priors for even faster, more informed decisions.

THANK YOU

Presented By Aman Anand
(CS22B054)

Github Repo Link :
https://github.com/AMAN9876543210/CS22B054_AI_3/