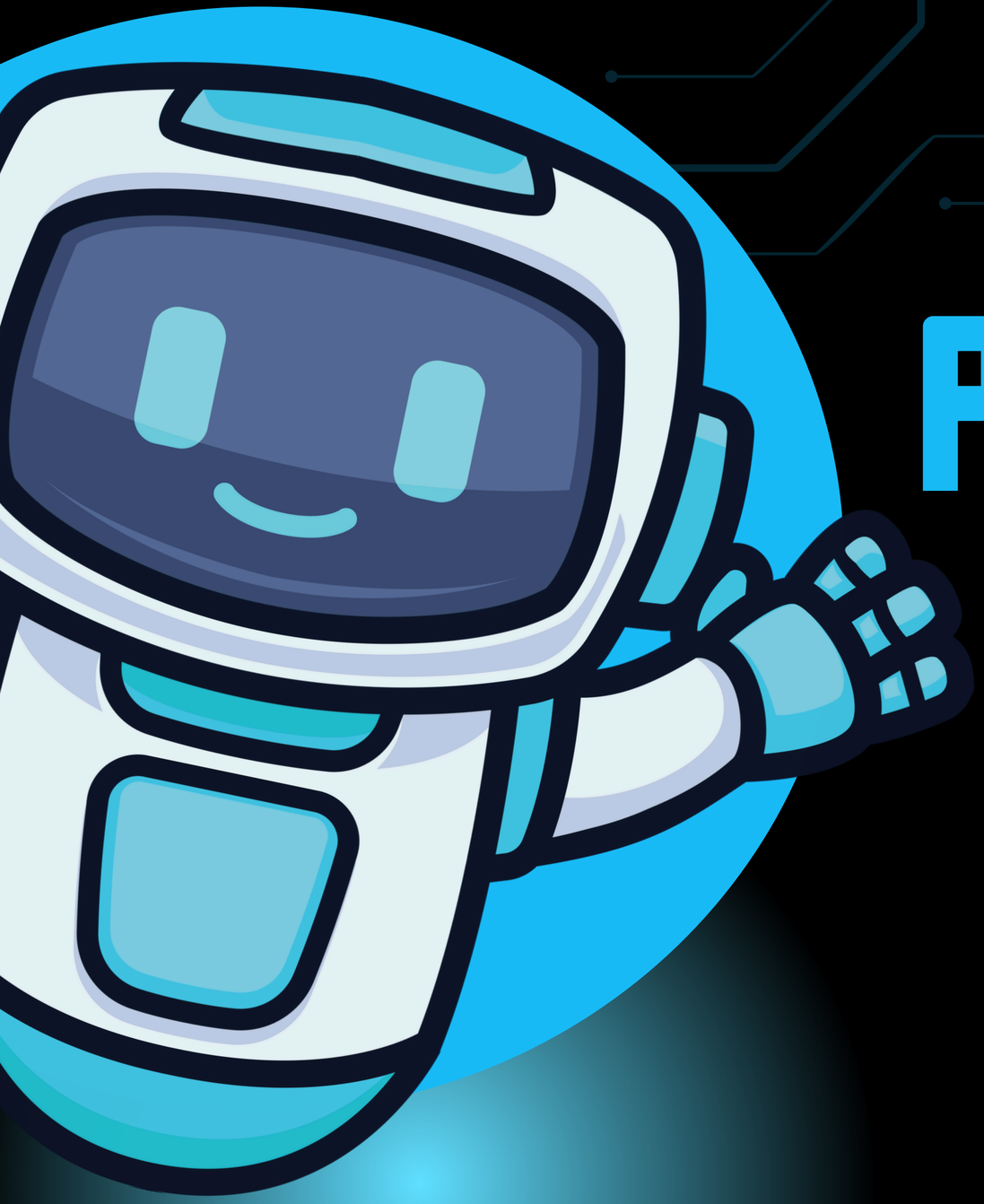

Presented By
Giovanni Stea
Şimal Yücel

PURRFORMANCE

FAIKR - Module 1 - NAO

THE LOGIC

- **The "Clumping" Penalty (+50 Cost):**
  - The Logic: The code detects if the robot tries to perform two "Mandatory Moves" (like Sit and Wipe Forehead) back-to-back.
  - The Goal: We force the robot to "space out" the required tasks. It prevents the robot from getting all the boring work done in the first 20 seconds. It forces it to mix "vegetables" (mandatory moves) with "dessert" (dance moves).
- **The "Repetition" Constraint (Infinite Penalty):**
  - The Logic: The robot is strictly forbidden from repeating the exact same move twice in a row (e.g., Wave -> Wave).
  - The Goal: Prevents the robot from looking like it is "glitching" or stuck in a loop.
- **The "Boredom" Mechanic:**
  - The Logic: Every move has a "Desire Score." If the robot executes a cool move (like The Robot), that move's desire score drops to zero immediately.
  - The Goal: This penalizes spamming. The robot is mathematically encouraged to wait for the audience's "boredom" to rise before doing that move again to get maximum points.

THE COST FUNCTION (G)

- **Standard A:*** Usually, A* tries to minimize cost (distance or time).
- **Our Logic:** We modified the cost function (path_cost) to prioritize style over speed.
- **Time Cost:** Every move costs time (seconds).
- **Aesthetic Reward:** We subtract "cost" for cool moves. High-energy moves (like Disco or TheRobot) have negative costs, effectively "rewarding" the algorithm for choosing them.
- **Boredom Penalty:** The code tracks a "boredom score." If a move isn't used for a while, it becomes more attractive. If used recently, it is "expensive" to repeat.

THE HEURISTIC (H)

- **Role:** The Heuristic guides the A* search by estimating how far we are from the goal.
- **The Formula: h(n)** = (Time needed for remaining mandatory moves) + (Number of pending moves × 20).
- **The "Pressure"**: By multiplying the pending moves by 20, we create a high "cost" for procrastinating. This forces the robot to prioritize clearing its "Mandatory List" rather than just dancing aimlessly.
- **Pruning:** If Time Elapsed + Estimated Time Left > 110 seconds, the heuristic returns "Infinity." This instantly kills that branch of the search tree, saving computing power by ignoring impossible plans.

STATE MANAGEMENT

- **The State:** At every step, the robot remembers four things:
  - The move it just finished.
  - Which "Mandatory Moves" are left to do.
  - How much time has passed.
  - The current "Aesthetic/Boredom" scores for all moves.
- **Physical Constraints:** The code uses a "compatibility graph." The robot cannot go from Sitting directly to Dancing; it must pass through Stand or StandZero. The algorithm only explores physically possible transitions.


THANKS
FOR
LISTENING!