Task 20: Research Plan

1. Cover Page

Student details: Kha Anh Nguyen – 103814796

Abstract:

The current *Dinner AutoDash* game successfully simulates a restaurant with a single autonomous agent using a combination of AI techniques. However, it lacks statistics and is limited to one staff member. This research plan outlines the steps to extend the project to investigate how staffing levels impact performance. The goal is to evolve the prototype into a research tool to answer a specific question: Is adding more AI agents always better? This plan details a structured approach to answering that question by comparing different system configurations and collecting quantitative data.

2. Key Research Question(s)

The core of this research is to explore the detailed impact of scaling the number of AI agents while verifying the AI for games techniques & logic are correctly implemented.

Primary Research Question:

*How does **scaling** the number of autonomous servo **agents** from one to three **affect** the **performance of the GOAP planning** and **A* pathfinding systems,** as measured by operational efficiency and customer satisfaction in a simulated restaurant?*

Sub-questions:

1. **Impact on Service Quality & Customer FSM:**

   How does an increase in the number of servos affect key customer-facing metrics like average wait time and final satisfaction rate?

   Rationale: The **CustomerFSM** logic can be tested using the statistics of real-world effectiveness by measuring its direct impact on the customer experience

2. **Impact on Economic Efficiency (GOAP Planner Effectiveness):**

   What is the net profit difference when running with one, two, or three servos, after accounting for staff wages and service penalties/rewards?

Rationale: This measures the effectiveness of the **ServoGOAPPlanner**. A higher profit indicates the planner is making economically optimal decisions beyond simply completing tasks.

3. **Impact on AI System Performance (A and GOAP Scalability):**

   What is the computational cost (average CPU time per tick) of increasing the number of servo agents, and does this suggest a point of diminishing returns for the AI systems?

   Rationale: This directly measures the scalability of the core AI algorithms (**pathfinder.py** and **goap_servo.py**). It assesses whether the benefits of adding more agents are eventually outweighed by the increased computational load.

3. Research Method

3.1 Experimental Platform

- **Language & Engine:** Python 3.9, Pygame 2.0+
- **Grid & Layout:** 10 columns × 8 rows; six tables fixed in a 3×2 arrangement.
- **Customers:** One "type" of customer. They spawn every 5 ticks (uniform rate) and follow the built-in FSM:
  WAITING → UNHAPPY (>=10 ticks wait time) → ANGRY (>=20 ticks) → LEAVING (if ≥ 30 ticks)
  or, once seated: SEATED → ORDERED → EATING → LEAVING.
- **Existing AI Modules:** The project already integrates Pathfinding (A*), Steering behaviors, a GOAP Planner, and a ServoAgent that ties them all together.

3.2 Profit & Cost Model

I will enhance world.py to track "net profit" at runtime:

1. **Initial capital:** $500 (set self.profit = 500 in *World.__init__*).
2. **Staff wage cost:** Each servo costs $20 per 60 ticks (i.e. $20 per "in-game hour").
3. **Angry-Leaving Penalty:** Whenever a customer reaches wait_time ≥ 30 and transitions to LEAVING (satisfaction → 0), subtract $30 immediately:

4. **Successful Meal Reward + Bonus:** Whenever a customer finishes EATING (eating_time ≥ 10) and transitions to LEAVING, add $30 plus a happiness bonus (HAPPY: +10, UNHAPPY: +5, ANGRY: No bonus)
5. **Display Profit in GUI:** In *world.drawAll()*, render text showing current profit

3.3 Enabling Multiple Servos

- The World constructor will be modified to accept a *num_servos* parameter.
- The GOAP and Servo agent logic will be adjusted to ensure each servo can independently compute plans and claim tasks without conflict.
- The rendering loop in *world.drawAll()* will be updated to draw all active servos.

3.4 Customer data storing:

Theorically, I will create a script (batch_run.py) which automates the experiment.

- **Loop over:** num_servos = 1, 2, and 3.
- **Trials:** 30 trials per condition (using seeds 0-29) to ensure statistical reliability.
- **Runtime:** Each trial will run for 250 ticks.
- **Metrics stored in results.csv:** To answer the sub-questions, the following quantitative data will be collected:
    1. Seed
    2. num_servos
    3. Average Wait Time
    4. Customer Satisfaction Rate (% of customers who don't leave angry)
    5. Net Profit at the end of the simulation
    6. Average CPU Time per Tick (ms)

3.5 Data Processing & Presentation

1. I will **load results.csv** into Python/pandas

2. And **compute group statistics** (mean ± stddev) for each metric, grouping by num_servos

3. **Statistical Tests**
    1. Perform two-sample t-tests (α = 0.05) comparing servos numberss on:
        1. avg_wait
        2. avg_sat
        3. profit
        4. CPU Time per Tick (track game performance)
4. **Charts** (Matplotlib or pandas plotting):
    1. **Net Profit**: one bar for 1 servo (mean ± std), one for 2 servos.
    2. **Average Wait Time** (mean ± std).
    3. **Average Final Satisfaction** (mean ± std).
    4. **Average CPU Time per Tick**.

    All charts include axis labels, legend, and captions.

4. Summary

This study will compare one-servo and two-servo configurations across 30 independent trials each. The detailed code implementation will be documented in **Task 21**. I will:

1. Extend the simulation with a profit model (initial capital $500; staff wage $20 per 60 ticks; $30 penalty for angry-leaving; $30 + bonus on meal completion).
2. Adapt the GOAP planner so that multiple servos can each invoke *compute_plan()* and reserve tables independently.
3. Instrument the code to record spawn, seating, finish ticks and profit.
4. Compute, for each trial, average wait time, average final satisfaction, net profit, and average CPU time per tick.
5. Present group statistics (mean ± std dev), bar charts with error bars, and two-sample t-tests to determine significance.

The results will quantify whether deploying a second servo leads to statistically significant gains in service metrics and profitability.

5. References

This research is guided by established work in the field of multi-agent systems and restaurant simulation. The following papers provides guidance for the methodology and goals of this project.

1. Multi-Agent Simulation-Based Analysis for Restaurant Service - (Nzinga René et al.)

- **Annotation:** This paper uses a multi-agent simulation (MAS) to model the interactions between customers, waitpersons, and kitchen staff to optimize restaurant efficiency and customer satisfaction. Its focus on agent-based modeling to analyze service processes is directly aligned with the core methodology of *DinnerAutoDash*.

2. Simulation and optimization of service system for restaurant - (Wang & Shi)

- **Annotation:** This study uses discrete event simulation to identify and solve "bottle-neck problems" in a restaurant's service system. This directly relates to this project's primary goal of testing how different numbers of servo agents (staff) affect system throughput and finding the optimal configuration.

3. Computer simulation: an important tool in the fast-food industry - (Kharwat)

- **Annotation:** This paper establishes the value of using computer simulation for "what if" analysis of staffing levels, workflow, and customer service in a major fast-food chain. It provides a strong, industry-backed precedent for the methods used in this research plan.

4. Discrete event simulation for quick service restaurant traffic analysis - (Jaynes & Hoffman)

- **Annotation:** This work describes a simulation tool designed to analyze and optimize traffic flow and site design for Quick Service Restaurants (QSRs). While its focus is on external traffic, it validates the use of simulation to perform "objective, quantitative comparisons between multiple arrangements of a site", which is the same principle this project applies to internal operations.

5. Simulation of restaurant operations using the Restaurant Modeling Studio - (Brann & Kulick)

- **Annotation:** This paper details a "virtual test bed" for investigating how factors like **staffing levels and physical layout** impact critical performance metrics in restaurants. Its method of running comparative analyses with different labor deployments to determine the impact on sales and service speed is a direct model for the experimental design of this research.

6. AutoDiner: Empowering Restaurant Simulations with Advanced LLMs for Enhanced Agent Collaboration - (Anonymous ACL submission)

- **Annotation:** This recent work introduces a simulator where multiple agents (chef, waiter, customer) collaborate to **maximize profitability and customer satisfaction**. This aligns perfectly with the research goals of *DinnerAutoDash* and demonstrates that multi-agent simulation is a current and important area of AI research.