ARTU: A Miniature AI-Assisted Robot with OLED Expressions and Object Detection

Bringing Intelligence and Expression to Your Desktop Companion!

INTRODUCTION

Objective

The objective of this project is to develop a Desk Pet, an AI-assisted miniature robotic companion that autonomously navigates a desk while detecting and avoiding obstacles. Additionally, the robot features object detection and pickup using ESP32-CAM, along with an OLED display that shows expressive animations. The Desk Pet can interact with users through Wi-Fi control and a touch sensor, triggering visual expressions and sound responses. This project aims to create a smart, interactive, and cost-effective robotic assistant that combines autonomous mobility, AI-based object detection, and engaging user interaction.

Concept

This AI-powered Desk Pet uses ultrasonic and IR sensors to detect obstacles and prevent falling off the desk. It is controlled by an Arduino Uno for autonomous movement, while NodeMCU handles Wi-Fi communication and OLED expressions. An ESP32-CAM enables object detection and pickup, allowing the robotic arm to identify and transport objects (e.g., a sharpener) to a fixed location. A touch sensor activates specific OLED eye expressions and sound effects using an ISD1820 voice module. The integration of AI-based detection, expressive OLED animations, and interactive control makes this robot a smart and engaging desktop companion.

Possible Applications

1. Interactive Desk Companion – A smart robotic pet that reacts to user interactions and moves autonomously.
2. AI-Based Object Handling – Detecting and picking up small objects with an intelligent robotic arm.
3. Educational Robotics – Teaching AI-based navigation, object recognition, and robotic manipulation.

CIRCUIT DIAGRAM

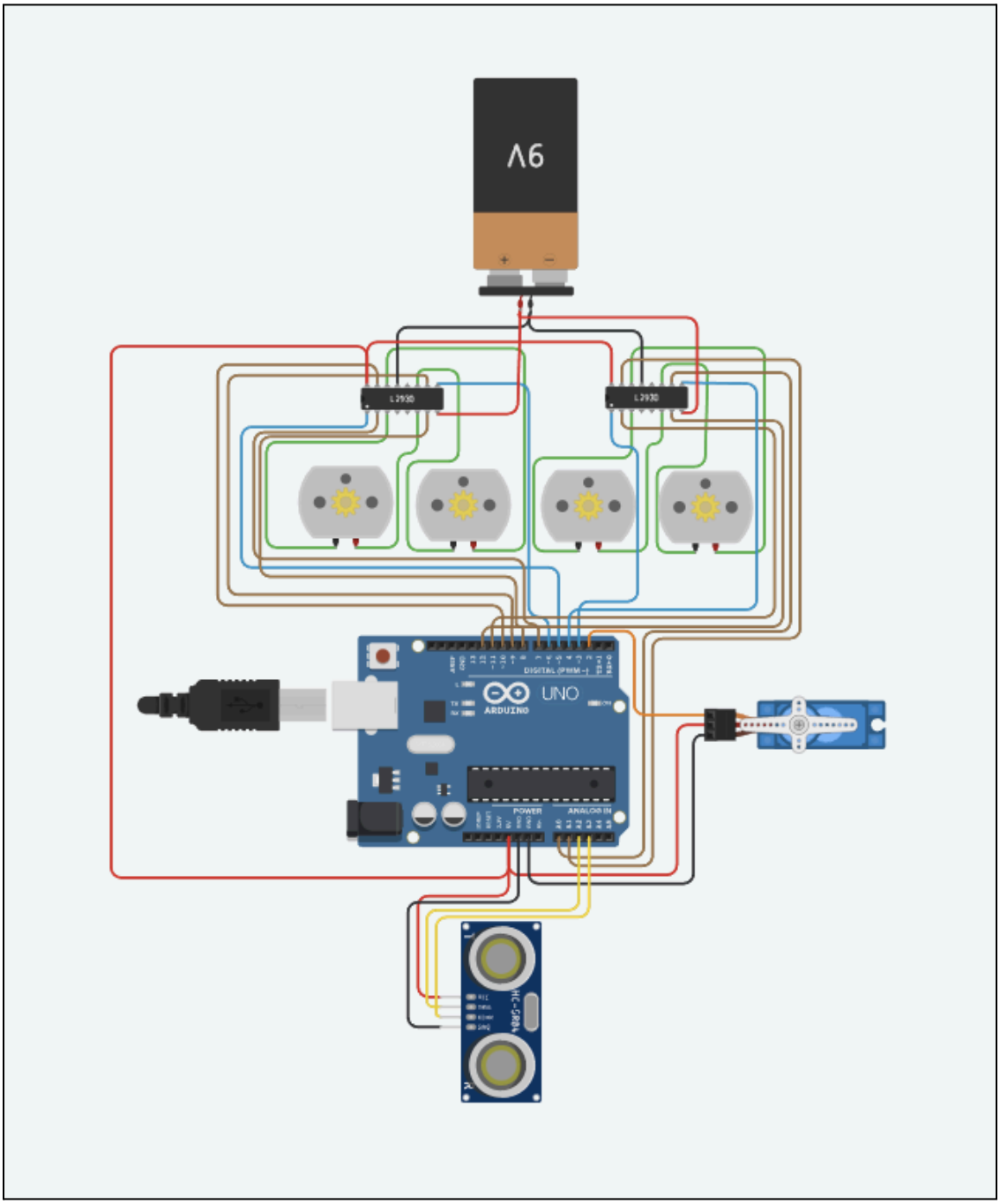

Fig 1 Circuit diagram of the desk pet

COMPONENTS

S.No.	Name	Description
1	Arduino Uno R3	Open-source microcontroller board used for controlling motors and sensors. Clock Speed: 16 MHz, Operating Voltage: 5V.
2	HC-05 Bluetooth	The HC-05 is a widely used, small, and versatile Bluetooth module that enables wireless serial communication (UART) between devices.
3	ESP32-CAM	AI-powered camera module for object detection and pickup. Features: Integrated Wi-Fi, OV2640 camera sensor.
4	L298N Motor Driver Module	Controls the movement of DC motors in the Desk Pet. Voltage Range: 5V – 35V, Current: Up to 2A per channel.
5	HC-SR04 Ultrasonic Sensor	Detects obstacles for autonomous navigation. Detection Range: 2 cm – 400 cm, Accuracy: ±3 mm.
6	IR Edge Detection Sensors	Prevents the robot from falling off edges by detecting surfaces. Operating Voltage: 3.3V – 5V.
7	SG90 Servo Motor (for robotic arm & ultrasonic sensor movement)	PWM-controlled servo motor for precise angle control. Rotation: 0° to 180°, Torque: 1.8 kg·cm.
8	N20 Gear Motors (for movement)	Small brushed DC motors for moving the Desk Pet. Speed: 300 RPM, Torque: 1-2 kg·cm.
9	4DOF Robotic Arm	A robotic arm integrated with the Desk Pet for object pickup using ESP32-CAM.
10	Rubber Wheels	Provides mobility; attaches to N20 gear motors for smooth movement.
11	ISD1820 Voice Module or 20 Ohm Speaker	Plays sound effects when the touch sensor is activated. Voltage: 3V – 5V.
12	OLED Display (128x32)	Displays animated eye expressions based on different robot states. Communication: I2C
13	Touch Sensor	Detects user interaction and triggers OLED eye changes & sound effects.
14	Jumper Wires	Used for connecting components. Available in various lengths (10cm, 20cm, etc.).
15	Lithium-ion (Li-ion) Battery (12V)	Provides power to the Desk Pet. Rechargeable options available for extended use.

Table 1: Components for Desk pet

LITERATURE SURVEY

Sl. No.	Author Name	Year of Publication	Algorithm	AI Feature	AI Implementation
1.	V. Mane, A. Patil, S. Patil and V. Patil	2025	A* Algorithm	ROS (Robot Operating System) Navigation Stack – Integrated for autonomous movement and planning.	AI-driven Simultaneous Localization and Mapping (SLAM) helps the robot navigate dynamically changing hospital environments without human intervention.
2.	Lu Wang, Lulu Liu, Xiaoxia Lu	2024	Particle Swarm Optimization	Uses pheromone concentration to improve pathfinding efficiency. Reduces unnecessary search areas, increasing convergence speed.	Enhances PSO by incorporating heuristic learning mechanisms to guide particles efficiently. Uses pheromone concentration techniques to refine search regions, preventing premature convergence.
3.	Faten Hamad, Hussam N. Fakhouri, Fawaz Alzghoul, Jamal Zraqou	2024	Gray Wolf Optimization	GWO for obstacle avoidance and optimal path planning	AI is used to improve path optimization efficiency, ensuring collision-free movement in dynamic environments while adapting to different scenarios.
4.	Min Zhuang, Ge Li, Kexin	2023	Artificial Potential Field	Combines APF's force	The hybrid AI model enhances

	Ding		(APF) Algorithm, A* Algorithm	field approach with A*'s shortest-path calculation to improve navigation efficiency.	convergence rate, reducing planning time while maintaining high accuracy.
5.	Yang Hong, Zhiyu Ding, Yuan Yuan, Wenzheng Chi, Lining Sun	2023	Conditional Variational Autoencoder (CVAE)	CVAE for learning dynamic obstacle avoidance strategies	AI is used to enable a robot to predict pedestrian movement and dynamically adjust its path for smooth and efficient navigation in crowded spaces.
6.	Ahtsham Alam, Syed Ahmed Abdullah, Israr Akhter, Suliman A. Alsuhibany, Yazeed Yasin Ghadi, Tamara al Shloul, Ahmad Jalal	2022	Convolutional Neural Networks (CNN) based object detection	AI-based object detection using image processing and ultrasonic sensors for autonomous navigation	AI is used for real-time object recognition and obstacle detection to enable self-driving vehicles to make intelligent navigation decisions and avoid collisions.
7.	Anuwat Angkuldee, Kuei-Ping Shih, Somchoke Ruengittinun	2019	A* algorithm	Pathfinding using A* in Webots simulation, with memory for detected obstacles	AI is used to enable robots to find optimal paths while avoiding obstacles efficiently, improving movement accuracy and decision-making.
8.	Ali Alyasin, Eyad I. Abbas, Sundus D. Hasan	2019	Dijkstra's Algorithm	Graph-based path planning using Raspberry Pi for shortest	AI is used to determine the most efficient route in road networks, minimizing travel

				path computation	time and cost for autonomous navigation.
9.	Rahib H. Abiyev, Murat Arslan, Irfan Günsel, Ahmet Çağman	2017	A* algorithm, SVM	SVM for obstacle classification and A* for path planning	AI is used to process images from a camera to classify areas as obstacle/non-obstacle and find the shortest path for a NAO humanoid robot.
10.	Fethi Belkhouche	2009	Virtual Plane Transformation, Collision Cones (CC), Dynamic Window Approach (DWA)	Virtual Plane method for transforming moving obstacles into stationary ones for path planning	AI is used to simplify dynamic obstacle avoidance by mapping a moving environment into a virtual plane where navigation decisions are easier.

Table 2: Literature Survey

AGENT AND ENVIRONMENT

1. Modularity

The system follows a hierarchical structure, with Arduino Uno managing motor control, NodeMCU handling Wi-Fi communication and OLED expressions, and ESP32-CAM performing object detection and pickup. Sensors like HC-SR04 and IR sensors assist in navigation and obstacle avoidance.

2. Planning the Horizon

The Desk Pet follows a continuous navigation approach, making real-time decisions based on sensor feedback and object detection.

3. Representation

The robot's state is determined by ultrasonic sensor readings (distance), IR sensor inputs (edge detection), motor activity (movement status), and ESP32-CAM object detection.

4. Computational Limits

The Arduino Uno and NodeMCU have limited processing power, relying on predefined movement algorithms, PWM control for motors, and ESP32-CAM's onboard object detection.

5. Learning

The system follows a rule-based approach rather than AI learning. However, future updates could integrate machine learning for improved object recognition and adaptive navigation.

6. Uncertainty

The Desk Pet encounters uncertainty due to sensor noise, variable lighting conditions for ESP32-CAM, and power fluctuations, affecting object detection and movement accuracy.

7. Preference

The system prioritizes safe navigation, object pickup, and interactive responses. It avoids obstacles, prevents falls, and reacts to touch sensor activation with OLED expression changes and sound effects.

8. Number of agents

A single-agent system, where the Desk Pet operates independently, processing sensor inputs, object detection, and user interactions without external AI agents.

9. Interactivity

The Desk Pet functions in real-time mode, continuously processing sensor data, object detection, and user interactions via Wi-Fi and touch sensor activation to adjust movement, pickup, and OLED responses dynamically.

10. Interaction with other dimensions

The system interacts across multiple dimensions, including physical (motor movement, robotic arm actions), sensory (ultrasonic, IR, and ESP32-CAM detection), and communication (Wi-Fi-based control, OLED expressions, and sound feedback). These layers work together for autonomous decision-making and user interaction.

ARCHITECTURE DIAGRAM

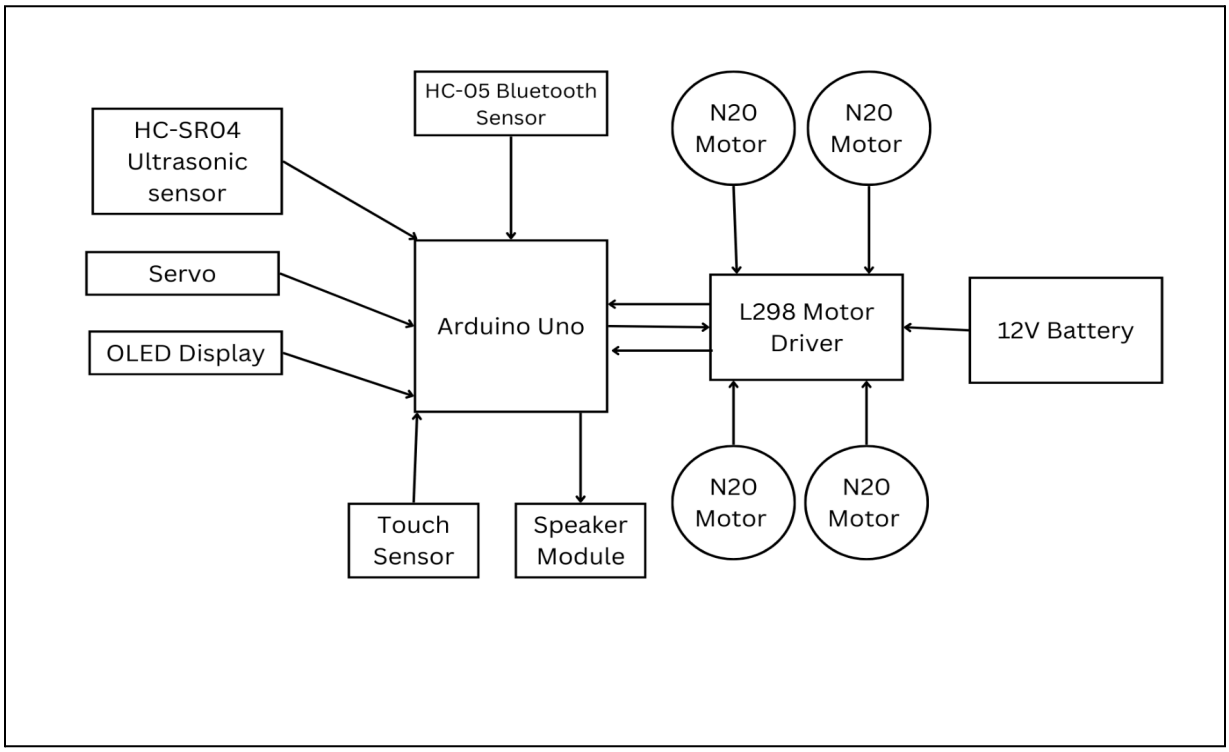

Fig 2 Architecture diagram of the desk pet

ALGORITHM

1. **A* Path Planning Algorithm for Desk Pet Robot**

A_STAR(start, goal)

1. Create an empty priority queue **OPEN** and insert start node.
2. Create an empty set **CLOSED**.
3. Set **g(start) = 0** and **f(start) = g(start) + h(start)**.
4. while **OPEN is not empty**:
   a. **current = node** with the lowest f(node) in **OPEN**
   b. If **current == goal**:
      return **RECONSTRUCT_PATH(current)**
   c. Remove **current** from **OPEN** and add to **CLOSED**.
   d. for each **neighbor** in **NEIGHBORS(current)**:
      If **neighbor** is in **CLOSED**, continue.
      tentative_g = g(current) + COST(current, neighbor)
      If **neighbor** not in **OPEN** or **tentative_g < g(neighbor)**:
      1. g(neighbor) = tentative_g
      2. f(neighbor) = g(neighbor) + h(neighbor)
      3. PARENT(neighbor) = current
      4. If **neighbor not in OPEN**, insert into **OPEN**
5. return **failure** (if no path found)

2. **Obstacle Avoidance Algorithm (Using Ultrasonic & IR Sensors)**

OBSTACLE_AVOIDANCE()

1. Read distance = HC_SR04()
2. Read left_edge, right_edge = IR_SENSORS()
3. If front_edge == detected OR rear_edge == detected:
   a. STOP_MOVEMENT()
   b. MOVE_BACKWARD()
   c. TURN_OPPOSITE_DIRECTION()
4. If distance < threshold:
   a. STOP_MOVEMENT()
   b. TURN_LEFT() or TURN_RIGHT() (random choice)
5. Else:
   a. MOVE_FORWARD()

3. **Object Detection & Pickup Algorithm (Using ESP32-CAM & Robotic Arm)**

OBJECT_DETECTION_AND_PICKUP()

1. Capture image using ESP32-CAM
2. Run Edge Impulse AI model on image
3. If "sharpener" detected:
   a. STOP_MOVEMENT()
   b. MOVE_ARM_DOWN()
   c. CLOSE_GRIPPER()
   d. MOVE_ARM_UP()
   e. NAVIGATE_TO_DROP_LOCATION()
   f. OPEN_GRIPPER() (Release sharpener)
   g. MOVE_ARM_TO_IDLE_POSITION()
4. Else:
   a. CONTINUE_MOVEMENT()

4. **OLED Expressions Algorithm (Based on Robot State)**

OLED_EXPRESSIONS(state)

1. If state == "running":
   DISPLAY("happy_eyes.gif")
2. If state == "stopped":
   DISPLAY("neutral_eyes.gif")
3. If state == "pickup":
   DISPLAY("focused_eyes.gif")
4. If state == "down":
   DISPLAY("tired_eyes.gif")
5. If state == "touch":
   DISPLAY("surprised_eyes.gif")

5. **Touch Sensor Interaction Algorithm**

TOUCH_SENSOR_EVENT()

1. If TOUCH_SENSOR() == activated:
   a. OLED_EXPRESSIONS("touch") # Change OLED to "surprised" eyes
   b. PLAY_SOUND("reaction.wav") # Play audio using ISD1820
   c. If holding object:
      RELEASE_OBJECT() # Drop the sharpener

6. **Bluetooth Control Algorithm (Using HC-05)**

   BLUETOOTH_CONTROL()

   1. Start HC-05 Bluetooth server
   2. If Wi-Fi command received:
      a. If command == "forward": MOVE_FORWARD()
      b. If command == "backward": MOVE_BACKWARD()
      c. If command == "left": TURN_LEFT()
      d. If command == "right": TURN_RIGHT()
      e. If command == "stop": STOP_MOVEMENT()

Complexity Analysis

Parameter	Explanation		Description
Time complexity	Best	O(d)	If the he-uristic function h(n) accurately estimates the cost to the goal, A* expands only nodes along the optimal path, leading to linear time complexity.
	Average	O(n log n)	In real-world scenarios, A* explores a subset of the search space, guided efficiently by heuristic evaluation. It performs similarly to Dijkstra's algorithm but is more focused.
	Worst	$O(b^d)$	When the heuristic is uninformative, A* expands nodes exponentially, depending on the branching factor (b) and depth (d) of the search tree.
Space Complexity	O(n)		A* stores all generated nodes, including the OPEN (priority queue) and CLOSED (visited nodes) lists, leading to high memory usage
Auxiliary Memory	O(n)		Extra space is needed for cost values, parent pointers (for path reconstruction), and priority queue management, making A* memory-intensive.
Stability	No		A* is not stable since it uses f(n) = g(n) + h(n) for cost evaluation. If two nodes have the same f-cost, their order depends on tie-breaking implementation.
Inplace	No		A* is not in-place as it requires additional memory structures for efficient pathfinding.

Table 3 Complexity analysis of the algorithm

TEST CASES & RESULTS

Category	Test Scenario	Input Condition	Output
Obstacle Detection	No Obstacle	Ultrasonic sensor detects >30 cm	Robot moves freely
Obstacle Detection	Near Obstacle	Ultrasonic sensor detects 10-30 cm	Robot slows down, OLED shows Scared
Obstacle Detection	Immediate Obstacle	Ultrasonic sensor detects <10 cm	Robot stops/turns, OLED shows Scared
Edge Detection	No Edge Detected	IR Sensor HIGH	Robot moves normally
Edge Detection	Edge Detected	IR Sensor LOW	Robot stops/moves back, OLED shows Surprised
Motor Control	Move Forward	PWM signal to both motors	Both wheels rotate forward
Motor Control	Move Backward	Reverse PWM signals	Both wheels rotate backward
Motor Control	Turn Left	Left motor stops, right motor moves	Robot turns left
Motor Control	Turn Right	Right motor stops, left motor moves	Robot turns right
Motor Control	Stop	No power to motors	Both wheels stop
OLED Expressions	Running Mode	Robot is moving freely	OLED shows Normal Eyes
OLED Expressions	Obstacle Detected	Ultrasonic sensor detects object	OLED shows Scared Eyes
OLED Expressions	Edge Detected	IR sensor detects edge	OLED shows Surprised Eyes
OLED Expressions	Touch Sensor Activated	User touches sensor	OLED shows Happy Eyes

OLED Expressions	Object Pickup Mode	Robotic arm picking up object	OLED shows Focused Eyes
OLED Expressions	Object Placed	Robotic arm drops the object	OLED shows Happy Eyes
Object Detection	Object Not Detected	ESP32-CAM does not recognize object	No action, OLED shows Neutral Eyes
Object Detection	Object Detected	ESP32-CAM detects a Object	Robotic arm moves to pick up Object, OLED shows Focused Eyes
Object Pickup	Touch Sensor Pressed (Release)	User presses touch sensor while arm holds object	Robotic arm releases object, OLED shows Happy Eyes
Sound Module	Touch Sensor Activated	User presses touch sensor	Play sound effect

Table 4 Test cases

CONCLUSION & DISCUSSION

This desk pet project demonstrates the potential of robotics in everyday interactions, offering a blend of functionality, education, and entertainment. By combining obstacle avoidance with an interactive oled expression display, it serves as a smart companion capable of engaging users in various settings. As technology advances, the desk pet can evolve with improved AI, increased autonomy, and expanded functionalities, making it a valuable addition to personal and professional environments.

The development of a desk pet with obstacle avoidance and integration of expression presents an exciting intersection of robotics, AI, and human interaction. By integrating advanced sensors and intelligent decision-making, the desk pet can navigate its environment seamlessly while engaging users in a meaningful way. The inclusion of expressions further enhances its versatility, allowing it to interact with humans.