Introduction to the Oracle Orb: A Fortune-Telling Robot

The **Oracle Orb** is a unique fusion of creative design and robotic engineering that brings the magic of fortune-telling to life. Imagine a whimsical robot that gracefully spins and pivots before revealing your fortune on a glowing screen. Inspired by traditional fortune-telling devices and powered by modern robotics, the Oracle Orb adds an enchanting twist to your DIY robotics projects. This build combines microcontroller programming, and mechanical design to create a mystical experience.

Whether you're an enthusiast of robotics, electronics, or simply enjoy bringing imaginative ideas to life, the Oracle Orb is the perfect project. With its smooth pivoting motion, interactive lighting effects, and fortune-telling capabilities, this project seamlessly blends mechanical, electrical, and software components into a fun and rewarding build.

Project Concept

Imagine pressing a button to awaken the Oracle Orb. The sphere begins to pivot slowly on its base. As the anticipation builds, the Orb spins more dynamically until it gradually comes to a stop. Then, a mystical fortune appears on the embedded LCD screen, delivering a personalized prediction. This magical moment is driven by precisely coordinated motor movements, and a carefully crafted fortune-telling algorithm.

The Oracle Orb isn't just a cool robotic gadget—it's an interactive experience where mechanical precision meets the mystical art of fortune-telling.

Summary of Components

Here's a breakdown of the components you'll need to bring the Oracle Orb to life:

- **Arduino Uno**: The core microcontroller, handling all the motor control,and fortune display.
- **Breadboard**: A platform for prototyping your circuit connections.
- **Jumper Wires**: Used for wiring the Arduino to the components.
- **Two Servo Motors (MG996R or stronger)**: These provide the rotational motion to the sphere.
- **Custom Pivot Mount**: Allows the sphere to rotate smoothly on two axes for an elegant pivoting effect.
- **Screws, Nuts, Bolts**: Basic hardware for assembling and securing the robot's components.
- **Sphere (5-6 inches)**: The main visual element, housing the LCD screen and rotating to reveal your fortune. **(Custom Made)**

- **Base (6-7 inches)**: Provides structural support for the entire device, including motors and wiring. **(Custom Made)**
- **Button or Switch**: Triggers the fortune-telling sequence, starting the Orb's movement.
- **LEDs and Resistors (optional)**: Enhances the visual appeal by illuminating the Orb during its motion.
- **Power Source**: Powers the Arduino, and motors (e.g., a 9V battery or regulated power supply).
- **Glue, Mounting Tape, or Wire Management Tools**: Keeps your components securely in place and organized.
- **Wire Strippers/Cutters and Screwdriver**: Essential tools for preparing wires and assembling the robot.

This list of components will set you up for a successful Oracle Orb build. In the next sections, we'll dive deeper into the step-by-step instructions to assemble, program, and bring this mystical fortune-telling robot to life.

Part 1: Wiring and Testing the LCD Component

In this step, we will wire the LCD1602 display to the Arduino Uno, using a breadboard to organize the connections. During this stage, the power will come from the computer via the USB connection to the Arduino.

Components List:

- **Arduino Uno**: To control the LCD.
- **LCD1602 Display**: To display text messages.
- **Breadboard**: For organizing the connections and distributing power and ground.
- **Jumper Wires**: To connect components on the breadboard and to the Arduino.
- **10kΩ Potentiometer**: For adjusting the contrast of the LCD display.
- **USB Cable**: To power the Arduino from the computer.

LCD1602 Pin Layout and Connections

Follow these instructions to connect the LCD to the Arduino:

1. **Connect Power and Ground for the LCD**:
    - Connect VSS (Pin 1) to the GND rail on the breadboard.
    - Connect VDD (Pin 2) to the 5V rail on the breadboard.
2. **Wiring the Potentiometer**:
    - Connect the middle pin of the potentiometer to V0 (Pin 3) on the LCD for contrast adjustment.
    - One outer pin of the potentiometer goes to the GND rail.
    - The other outer pin goes to the 5V rail.
3. **Connect Control Pins**:

    - Connect RS (Pin 4) to Digital Pin 12 on the Arduino.
    - Connect RW (Pin 5) to the GND rail to set it to write mode.
    - Connect E (Pin 6) to Digital Pin 11 on the Arduino.
4. **Connect Data Pins (4-bit mode)**:
    - Connect D4 (Pin 11) to Digital Pin 5 on the Arduino.
    - Connect D5 (Pin 12) to Digital Pin 4 on the Arduino.
    - Connect D6 (Pin 13) to Digital Pin 3 on the Arduino.
    - Connect D7 (Pin 14) to Digital Pin 2 on the Arduino.
5. **Backlight Wiring**:
    - Connect A (Pin 15) to the 5V rail.
    - Connect K (Pin 16) to the GND rail.
6. **Optional Resistor**: If the backlight is too bright, place a 220Ω or 330Ω resistor between Pin 15 (A) and the 5V rail.

Testing the LCD

Now that the wiring is complete, it's time to test the LCD with a simple program to ensure that it displays text properly.

```
#include <LiquidCrystal.h>

// Initialize the library with the numbers of the interface pins
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
  // Set up the LCD's number of columns and rows
  lcd.begin(16, 2);

  lcd.print("Oracle Orb");
  delay(30000);
  lcd.clear();
}

void loop() {
  lcd.setCursor(0, 0);

  lcd.print("Fortune awaits");

  delay(5000);
}
```

Instructions for Testing:

1. **Upload the code**: Copy the code into the Arduino IDE and upload it to your Arduino.
2. **Power the Circuit**: The Arduino will be powered through the USB cable connected to your computer.
3. **Adjust the Contrast**: Use the potentiometer to adjust the contrast of the display. Turn the knob until the text is clear.
4. **Check the Display**: You should see "Oracle Orb" on the first line and "Fortune awaits" on the second line.

With the LCD working and connected, you're ready to move on to testing the servo motors in the next part of the tutorial.

Part 2: Wiring and Testing a Single Servo Motor

Now that your LCD display is set up, the next step is to wire and test a single servo motor. This test will help ensure that the servo motor functions correctly and responds as expected. The servo motor will be powered through the Arduino, which will be connected to the computer for this test.

Components List:

- **Arduino Uno**: To control the servo motor.
- **Servo Motor (MG996R or similar)**: Provides the rotational motion.
- **Breadboard**: For organizing the connections.
- **Jumper Wires**: To connect components on the breadboard and to the Arduino.
- **USB Cable**: To power the Arduino from the computer.

Wiring the Servo Motor

1. **Connect the Servo Power and Ground**:
    - Connect the **red wire** from the servo motor to the 5V pin on the Arduino.
    - Connect the **black or brown wire** (ground) from the servo motor to the GND pin on the Arduino.
2. **Connect the Control Wire**:
    - Connect the **yellow or white wire** (signal) from the servo motor to Digital Pin 9 on the Arduino.

Summary of Connections:

- **Servo Red Wire (Power)** → 5V on Arduino
- **Servo Black or Brown Wire (Ground)** → GND on Arduino
- **Servo Yellow or White Wire (Signal)** → Digital Pin 9 on Arduino

Testing the Servo Motor

Now that the servo motor is wired up, you can run a simple test to verify that it works as expected. This test will rotate the servo to different angles.

```
#include <Servo.h>

Servo myServo;  // Create a Servo object to control the servo

void setup() {
  // Attach the servo on pin 9 to the Servo object
  myServo.attach(9);
}

void loop() {
  // Move the servo to 0 degrees
  myServo.write(0);
  delay(1000);  // Wait for 1 second

  myServo.write(90);
  delay(1000);

  myServo.write(180);
  delay(1000);

  myServo.write(90);
  delay(1000);
}
```

Instructions for Testing:

1. **Upload the Code**: Copy the code into the Arduino IDE and upload it to your Arduino.
2. **Power the Circuit**: The Arduino will be powered through the USB cable connected to your computer.
3. **Observe the Servo Motor**: The servo should rotate to 0°, then 90°, then 180°, and back to 90°, pausing for 1 second at each position.

Expected Behavior:

- The servo motor should smoothly rotate between the angles specified (0°, 90°, and 180°), pausing for 1 second at each position before moving to the next.

Once the servo motor works as expected, you're ready to proceed with more complex movements or multiple servos in the next stages of the Oracle Orb project.

Part 3: Creating the Shell for the Oracle Orb

In this step, you'll begin creating the base and sphere that will hold the components of the Oracle Orb. You have two options: 3D printing the provided CAD models or manually replicating the design. After constructing the shell, you'll test the fit of the components and begin organizing the wires inside the structure.

Components List:

- **Provided CAD Models**: For the base and sphere.
- **3D Printer** (optional): For printing the CAD models.
- **Alternative Materials** (optional): For manual replication (e.g., Styrofoam, plastic, wood).
- **Cutting Tools**: If replicating manually.
- **Glue or Tape**: For temporarily securing parts.
- **Screws**: For securing the base in future steps.

Step 1: Create the Base and Sphere

Option 1: 3D Printing the CAD Models

1. **Prepare the 3D Printer**:
    - Load the provided CAD files into slicing software (e.g., Cura or PrusaSlicer).
    - Ensure the printer settings match your material (PLA or ABS recommended).
2. **Start the 3D Printing Process**:
    - Begin printing the base and sphere sections. This will likely take several hours depending on the complexity and size.
    - Monitor the print to ensure it progresses smoothly without issues.
3. **Post-Printing**:
    - Once printing is complete, carefully remove the pieces from the printer bed.
    - Use sandpaper or a file to smooth any rough edges, particularly around snap-in sections.

Option 2: Manually Replicating the Design

1. **Choose Your Materials**:
    - Select sturdy materials like Styrofoam, wood, or plastic for the base and sphere.
2. **Cut and Shape the Pieces**:
    - Use the CAD models as a guide to trace the design onto your material.
    - Cut out and shape the pieces according to the model's dimensions.
3. **Assemble the Pieces**:
    - Use glue or tape to temporarily secure the base and sphere components.
    - Ensure the snap-in sections and removable base top are correctly replicated.

Step 2: Test the Fit and Organize Wires

1. **Place the Components Inside**:
    - Insert the Arduino, LCD screen, and servo motors into the base and sphere.
    - Ensure all components fit securely and that openings for the LCD and motors are aligned properly.
2. **Organize the Wires**:
    - Route the wires from the Arduino, LCD, and motors to ensure they do not obstruct moving parts.
    - Plan the pathways for organizing the wires during assembly, keeping them neat and out of the way.

Step 3: Preliminary Base Setup and Snap-in Test

1. **Align the Base**:
    - Temporarily align the top and bottom sections of the base to confirm they fit together securely.
    - Make sure the top is removable for future access to the components.
2. **Test Snap-in Features**:
    - Snap the two halves of the sphere together to ensure that the snap-in mechanism functions smoothly and can be easily disassembled and reassembled.
3. **Prepare for Final Securing**:
    - While full base securing will be completed in the next step, ensure that all parts can be easily assembled and taken apart without shifting the components inside.

Expected Outcomes:

- **Base and Sphere Created**: The base and sphere should now be physically constructed, either 3D printed or manually replicated.
- **Fit Test Complete**: All components should fit securely inside the shell, and the snap-in features should work as intended.
- **Wiring Map Created**: You should have a clear plan for routing and securing the wires inside the shell.

With the base, sphere, and snap-in features successfully tested, you'll be ready to proceed to the next step, which involves securing the base and finalizing the assembly.

Part 4: Uploading the Arduino Code for Continuous Rolling Effect

In this step, we will load the updated Arduino code that makes the Oracle Orb's sphere roll continuously in one direction for 20-30 seconds. The code will trigger when a button is pressed, after which the sphere will roll, and a random fortune will be displayed on the LCD screen. This

stage focuses on preparing the Arduino by uploading the code and ensuring it compiles without errors.

Components List:

- **Arduino Uno**
- **USB Cable** (to upload the code)

Step 1: Code Explanation

The updated code initializes two continuous servo motors, detects when the button is pressed, makes the sphere roll continuously for a random duration between 20-30 seconds, and then displays a randomly selected fortune on the LCD screen. The code will only execute when the button is pressed.

```
#include <Servo.h>
#include <LiquidCrystal.h>

// Define button and servo pins
const int buttonPin = 8;  // Button pin
Servo servo1;  // Continuous Servo motor 1
Servo servo2;  // Continuous Servo motor 2
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);  // Initialize LCD

// List of fortunes
String fortunes[] = {
  "Yes",
  "No",
  "Maybe",
  "Ask Later",
  "It’s Unclear",
  // Add more fortunes (up to 20-30)
};
const int numFortunes = sizeof(fortunes) / sizeof(fortunes[0]);

void setup() {
  // Initialize the button pin
  pinMode(buttonPin, INPUT_PULLUP);  // Button pin with pull-up resistor

  // Initialize the servos
  servo1.attach(9);
  servo2.attach(10);
```

```
  // Initialize the LCD and display the welcome message
  lcd.begin(16, 2);
  lcd.print("Oracle Orb");
}

void loop() {
  // Check if the button is pressed
  if (digitalRead(buttonPin) == LOW) {
    // Clear the LCD once the button is pressed
    lcd.clear();

    // Perform the continuous rolling effect for 20-30 seconds
    performContinuousRoll();

    // Pick a random fortune
    String fortune = pickRandomFortune();

    // Stop the servos and return to neutral position
    servo1.write(90);  // Stop motor
    servo2.write(90);

    // Display the selected fortune
    displayFortune(fortune);
  }
}

void performContinuousRoll() {
  // Simulate continuous rolling in one direction for 20-30 seconds
  int rollDuration = 20000 + random(0, 10001);  // Random time between
20-30 seconds
  unsigned long startTime = millis();

  while (millis() - startTime < rollDuration) {
    servo1.write(180);  // Full speed forward for servo 1
    servo2.write(0);    // Full speed forward for servo 2 (reverse
direction for balance)
  }

  // Stop the servos after the rolling effect
  servo1.write(90);  // Neutral position (stop)
  servo2.write(90);  // Neutral position (stop)
}
```

```
String pickRandomFortune() {
  // Select a random fortune from the list
  int index = random(0, numFortunes);
  return fortunes[index];
}

void displayFortune(String fortune) {
  // Display the selected fortune on the LCD
  lcd.setCursor(0, 0);
  lcd.print(fortune);
}
```

Step 2: Upload the Code to the Arduino

1. **Open the Arduino IDE**:
    - Launch the Arduino IDE on your computer and open a new sketch.
2. **Copy the Updated Code**:
    - Paste the above code into the new sketch.
3. **Connect the Arduino**:
    - Use the USB cable to connect the Arduino Uno to your computer.
4. **Select the Correct Board and Port**:
    - In the Arduino IDE, go to **Tools > Board** and select **Arduino Uno**.
    - Under **Tools > Port**, select the port corresponding to your Arduino.
5. **Upload the Code**:
    - Click the **Upload** button (the right-facing arrow in the IDE) to upload the code to the Arduino.
6. **Check for Compilation and Upload Errors**:
    - Ensure that the code compiles successfully and is uploaded without any errors.

Step 3: Verify the Upload

At this stage, you won't be able to fully test the motion since the wiring isn't complete, but you should ensure the following:

- The code uploads successfully to the Arduino Uno.
- The Arduino responds when connected to the computer.

Expected Outcomes:

- The code should compile and upload to the Arduino Uno without any errors.
- The Arduino will be ready to control the continuous rolling of the sphere once the wiring is completed and connected.

This step ensures the Arduino is fully prepared for controlling the rolling motion and displaying fortunes when wiring and assembly are finished.

Part 5: Wiring the Arduino, Two Servo Motors, and LCD Together

In this step, we will connect the Arduino, two continuous servo motors, and the LCD to a breadboard and power the entire circuit with a 9V battery. This will allow you to set up and test the circuit outside of the base and sphere before final assembly.

Components List:

- **Arduino Uno**
- **Two Continuous Servo Motors**
- **LCD (16x2) Display**
- **Breadboard**
- **Jumper Wires**
- **10kΩ Potentiometer** (for adjusting LCD contrast)
- **Button or Switch**
- **Resistor (10kΩ or 220Ω)** (for the button)
- **Transistor** (for motor control, optional for improved performance)
- **Diode** (for motor protection, optional)
- **9V Battery** and **Battery Clip** (to power the circuit)

Step 1: Wiring the LCD

The LCD will display the fortunes, and you need to wire it as follows:

LCD Pin Connections:

1. **VSS (Pin 1)** → GND on the breadboard
2. **VDD (Pin 2)** → 5V on the breadboard
3. **V0 (Pin 3)** → Middle pin of a 10kΩ potentiometer (for contrast adjustment)
    - One outer pin of the potentiometer → GND
    - The other outer pin → 5V
4. **RS (Pin 4)** → Digital Pin 12 on Arduino
5. **RW (Pin 5)** → GND on the breadboard
6. **E (Pin 6)** → Digital Pin 11 on Arduino
7. **D4 (Pin 11)** → Digital Pin 5 on Arduino
8. **D5 (Pin 12)** → Digital Pin 4 on Arduino
9. **D6 (Pin 13)** → Digital Pin 3 on Arduino
10. **D7 (Pin 14)** → Digital Pin 2 on Arduino
11. **A (Pin 15)** → 5V on the breadboard (Backlight Anode)
12. **K (Pin 16)** → GND on the breadboard (Backlight Cathode)

Step 2: Wiring the Servo Motors

The servo motors will control the rolling motion of the sphere. Each servo motor has three wires: power (red), ground (black or brown), and signal (yellow or white).

Servo Motor Connections:

1. **Servo 1**:
    - **Red Wire (VCC)** → 5V on the breadboard
    - **Black Wire (GND)** → GND on the breadboard
    - **Yellow Wire (Signal)** → Digital Pin 9 on Arduino
2. **Servo 2**:
    - **Red Wire (VCC)** → 5V on the breadboard
    - **Black Wire (GND)** → GND on the breadboard
    - **Yellow Wire (Signal)** → Digital Pin 10 on Arduino

Step 3:

```
Wiring the Button
The button will trigger the rolling action and fortune display.
Button Connections:
One side of the button → Digital Pin 8 on Arduino
Other side of the button → GND on the breadboard
10kΩ Resistor between Digital Pin 8 and 5V to pull the button signal up
when not pressed.
Step 4: Transistor and Diode (Optional)
For better control and protection of the servos, you can use a transistor
and diode:
Collector of NPN Transistor → Power line to the servo motors (Red Wire)
Emitter of Transistor → GND on the breadboard
Base of Transistor → Digital Pin 9 (or control pin) with a small
current-limiting resistor (e.g., 220Ω)
Diode connected across the motor's power terminals for voltage protection:
Cathode to the power line (5V)
Anode to the ground.
Step 5: Powering the Circuit with a 9V Battery
Connect the Battery:
Attach the 9V battery to the battery clip.
Connect the red wire from the battery clip to the VIN pin on the Arduino.
Connect the black wire from the battery clip to the GND pin on the Arduino.
This will provide power to the Arduino and the entire circuit via the VIN
pin, which regulates the voltage to 5V for the servos and LCD.
Step 6: Complete Wiring Diagram (Summary)
Breadboard Layout Summary:
```

Connect the 5V and GND rails on the breadboard to the 5V and GND pins on the Arduino.
Connect the LCD to its corresponding Arduino pins and power through the breadboard.
Connect Servo 1 and Servo 2 to Digital Pins 9 and 10 for control, and their power and ground to the breadboard.
Connect the Button to Digital Pin 8, with a 10kΩ resistor between the button and 5V.
Optionally, connect the Transistor and Diode for motor control and protection.
Power the entire circuit using the 9V battery connected to the VIN pin on the Arduino.
Testing the Setup
Upload the Code: Make sure the code from Part 4 is uploaded to the Arduino.
Power the Arduino: Connect the 9V battery to power the Arduino and the circuit.
Press the Button: When the button is pressed, the servos should start rotating continuously for 20-30 seconds, and the LCD should display a random fortune.
Expected Outcomes:
The LCD should display "Oracle Orb" upon powering up.
When the button is pressed, the servos should rotate continuously, simulating the rolling motion for 20-30 seconds.
After the rolling stops, the LCD should display a randomly selected fortune.
By powering the circuit with a 9V battery, the system becomes portable and can be easily integrated into the base and sphere later in the project.
Part 5: Final Assembly of the Oracle Orb (Including Button Placement)
In this final assembly step, you will combine all the components (Arduino, servos, LCD, button, etc.) into the physical shell. The button will be specially mounted at the top of the base for user interaction. Most components will sit in the base, the sphere will move with the servo motors, and the LCD will be mounted inside the sphere.
Components List for Assembly:
Assembled Circuit (Arduino, Servo Motors, LCD, Button, etc.)
Base: Holds the Arduino and other components.
Sphere: Moves with the servo motors and contains the mounted LCD screen.
Screws, Nuts, and Bolts: To securely mount components.
Wire Management Tools: Zip ties, mounting tape, etc.
Glue or Mounting Tape: For additional securing of parts.
Pivot Brackets or Mounts: For attaching the sphere to the servo motors.
Breadboard and Jumper Wires: For final wiring and connections.
Button: To be mounted on the top of the base for user access.

9V Battery and Battery Clip: For powering the system.

Step 1: Mounting Components in the Base

Mount the Arduino:

Place the Arduino in the bottom of the base and secure it using screws or mounting tape. Ensure the VIN pin is accessible for connecting the 9V battery.

Secure the Breadboard:

Place the breadboard inside the base next to the Arduino. This will connect the power and ground for the servos, LCD, and button.

Position the Servos:

Attach the two continuous servo motors to the base using pivot brackets or custom mounts. The motors should be positioned so that their shafts are aligned with the internal structure of the sphere, allowing for smooth rotation.

Connect the Battery:

Place the 9V battery in a secure spot inside the base and connect the battery clip to the VIN and GND pins on the Arduino.

Step 2: Mounting the Button on Top of the Base

Prepare the Button Mount:

Create a hole in the top of the base for the button, ensuring that it fits snugly and is easily accessible to the user.

Wire the Button:

One side of the button should be connected to Digital Pin 8 on the Arduino. The other side of the button should be connected to GND on the breadboard. Use a 10kΩ resistor to pull up the signal, connecting it between Digital Pin 8 and 5V to ensure stable readings when the button is not pressed.

Secure the Button:

Once wired, mount the button securely into the hole at the top of the base. Ensure the wiring is long enough to connect from the top of the base to the breadboard without causing tension.

Step 3: Mounting the LCD Screen in the Sphere

Prepare the Cut-Out:

Use the cut-out in the sphere to mount the LCD screen. Ensure that the screen sits flush with the surface and is visible when the sphere is in use.

Secure the LCD:

Mount the LCD screen inside the sphere using screws or glue. Ensure the LCD is securely held in place and the wiring can be routed through a small opening that leads into the base.

Step 4: Connecting the Sphere to the Servo Motors

Attach the Sphere to the Servo Motors:

Connect the sphere to the continuous servo motors using pivot brackets or a gimbal mount. Ensure the servos are securely attached to the internal

structure of the sphere, allowing the sphere to rotate smoothly.
Ensure Proper Alignment:
Test the attachment to make sure the sphere rotates without wobbling or resistance. Ensure the servos rotate the sphere in the desired direction for the rolling effect.
Step 5: Final Wiring and Connections
Route All Wires:
Connect all wiring from the LCD, button, and servos to the breadboard inside the base. Ensure all connections are secure, and the wiring is organized to prevent tangling or interference with the servo motion.
Organize and Secure Wiring:
Use zip ties or mounting tape to secure the wiring and prevent it from obstructing the movement of the sphere. Ensure the wiring does not get caught in any moving parts.
Step 6: Testing and Adjustments
Power On the Circuit:
Connect the 9V battery to power the system. The Arduino should start up, and the LCD should display the welcome message ("Oracle Orb").
Press the Button:
Test the system by pressing the button, which should trigger the rolling motion of the sphere. The servos should rotate the sphere for 20-30 seconds, and after that, the LCD should display a random fortune.
Make Adjustments:
If the sphere does not rotate smoothly, adjust the alignment of the servo motors or the mounting mechanism.
Ensure that the button is securely mounted and triggers the rolling action consistently.
Final Checks:
Sphere Rotation: The sphere should rotate smoothly in one direction for 20-30 seconds when the button is pressed.
Button Accessibility: The button should be easily accessible at the top of the base and trigger the rolling action reliably.
LCD Display: The LCD screen should display a random fortune once the rolling stops.
Wiring: All wiring should be secure and organized, with no loose or exposed wires.
Stability: The base should securely hold all components, and the sphere should rotate without wobbling or instability.
With the button securely mounted and the sphere rotating as intended, the Oracle Orb will be fully functional and ready for demonstration.