Sensor Measurement Assignment – Automatic Plant Watering (Indoor)

Gathering and interpreting sensor input data on a microcontroller (Arduino Uno). Sensor: Capacitive Soil Moisture Sensor v2.0 (AOUT→A0). A CAD plan is included for integration inside the 3D-printed elephant enclosure (hat = electronics bay; body = pump and tubing to trunk).

1. What am I measuring?

A proxy for volumetric soil moisture via the analog output (AOUT) of the capacitive probe. The Arduino Uno reads A0 and maps it to moisture % using a per-pot calibration (dry vs. field-capacity).

Figure 1: Capacitive Soil Moisture Sensor v2.0 (VCC, GND, AOUT).

2. Is the data what I expect?

More or less . Dry soil produces higher raw ADC values; wet soil produces lower values. With dry ≈ 790–820, wet ≈ 360–420. After trimmed-mean filtering of 20 reads, the trend follows watering and evaporation smoothly.

3. Do I have a sufficient sample rate?

For interactive tests: 1 Hz (every 1 s). For logging/operation: every 30–60 s, because soil moisture changes slowly and lower cadence saves power.
To make this project more accurate and more sensitive for the plant I was thinking to add a DHT11 (ambient) .

4. Outer bounds of the signal

Arduino ADC range is 0–1023 (0–5 V). Typical dynamic span observed with this sensor/pot is ~300–500 counts between dry and wet. Final bounds come from calibration constants RAW_DRY and RAW_WET.

5. What might affect signal quality (and mitigations)

Power ripple/USB noise and relay/motor EMI; cable length/orientation and inconsistent insertion depth; soil salinity/fertilizer changes; temperature. Mitigations: common ground, short/twisted pump leads, flyback diode across pump, local decoupling (0.1 µF + 100 µF at sensor), trimmed-mean averaging, depth mark on the probe.

6. How does the sensor detect the value? (brief)

The board implements a small RC oscillator whose effective capacitance rises with soil water content (water's dielectric ≈ 80). The oscillator shift is converted to an analog voltage on AOUT; the Uno reads this on A0. Being capacitive, it avoids electrolysis seen in resistive forks.

Wiring: Sensor VCC→5 V, GND→GND, AOUT→A0. Relay SRD-05VDC-SL-C: VCC→5 V, GND→GND, IN→D8; contacts COM/+5 V and NO→pump +; pump −→ supply GND (shared).
Sampling: trimmed-mean of 20 ADC readings per sample. Calibration per pot: (1) RAW_DRY after ~48 h dry; (2) RAW_WET ~30 min after watering; (3) map to % and verify monotonic trend over 24–48 h.

>For better testing I will incrementally add a LCD display

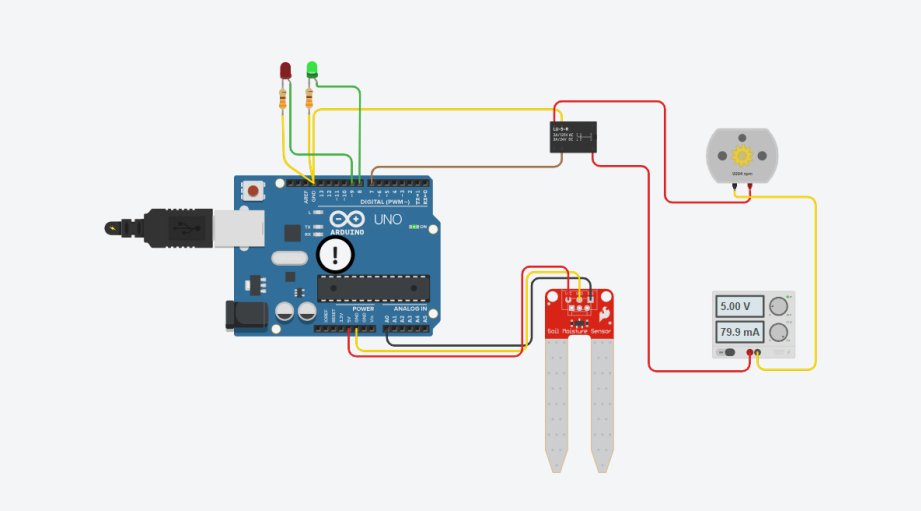

Figure 2: Tinkercad wiring – sensor A0, relay D8, status LEDs.

7. Minimal measurement sketch

```
const int SOIL=A0;
int RAW_DRY=800, RAW_WET=380;

int readFiltered(){const int N=20;int v[N];
  for(int i=0;i<N;i++){v[i]=analogRead(SOIL);delay(5);}
  for(int i=0;i<N;i++) for(int j=i+1;j<N;j++) if(v[j]<v[i]){int
t=v[i];v[i]=v[j];v[j]=t;}
  long s=0; for(int i=2;i<N-2;i++) s+=v[i]; return s/(N-4);}

int mapPct(int raw){float p=(RAW_WET<RAW_DRY)?100.0*(RAW_DRY-raw)/(RAW_DRY-RAW_WET)
                                  :100.0*(raw-RAW_DRY)/(RAW_WET-RAW_DRY);
  if(p<0)p=0; if(p>100)p=100; return (int)p;}

void setup(){Serial.begin(115200);}
void loop(){int r=readFiltered();int pct=mapPct(r);
  Serial.print("RAW=");Serial.print(r);Serial.print(" PCT=");Serial.println(pct);
  delay(1000);}
```

8. Results snapshot

Dry (48 h): RAW ≈ 790 | Field capacity (30 min): RAW ≈ 370 | Daily idle: RAW ≈ 520–560

9. CAD integration plan (elephant enclosure)

The probe extends from the pot rim to mid-height (approximately 2–3 cm from the wall), with its cable routed through a rear grommet into the elephant's body. The hat section serves as the electronics bay, housing the **Arduino Uno**, **relay**, **RTC**, **DHT11**, **LCD**, and **control buttons** on a 2–3 mm mounting plate with standoffs for stability. The elephant's body accommodates the **mini water pump** and **tubing**, with the outlet channel routed through the **trunk** for watering.

The 3D modeling workflow follows **MakerWorld Image-to-3D** for initial mesh generation and **Fusion 360** for refining geometry, splitting the parts, and adding mounting cavities.

Future Improvement: In future iterations, the design will include a **rotating turntable base** to allow the elephant to rotate, enabling irrigation of multiple nearby plants. This modification will also address current space constraints by concealing more of the electronic components inside the expanded base, as the hat section alone is too small to accommodate all hardware components neatly.

.

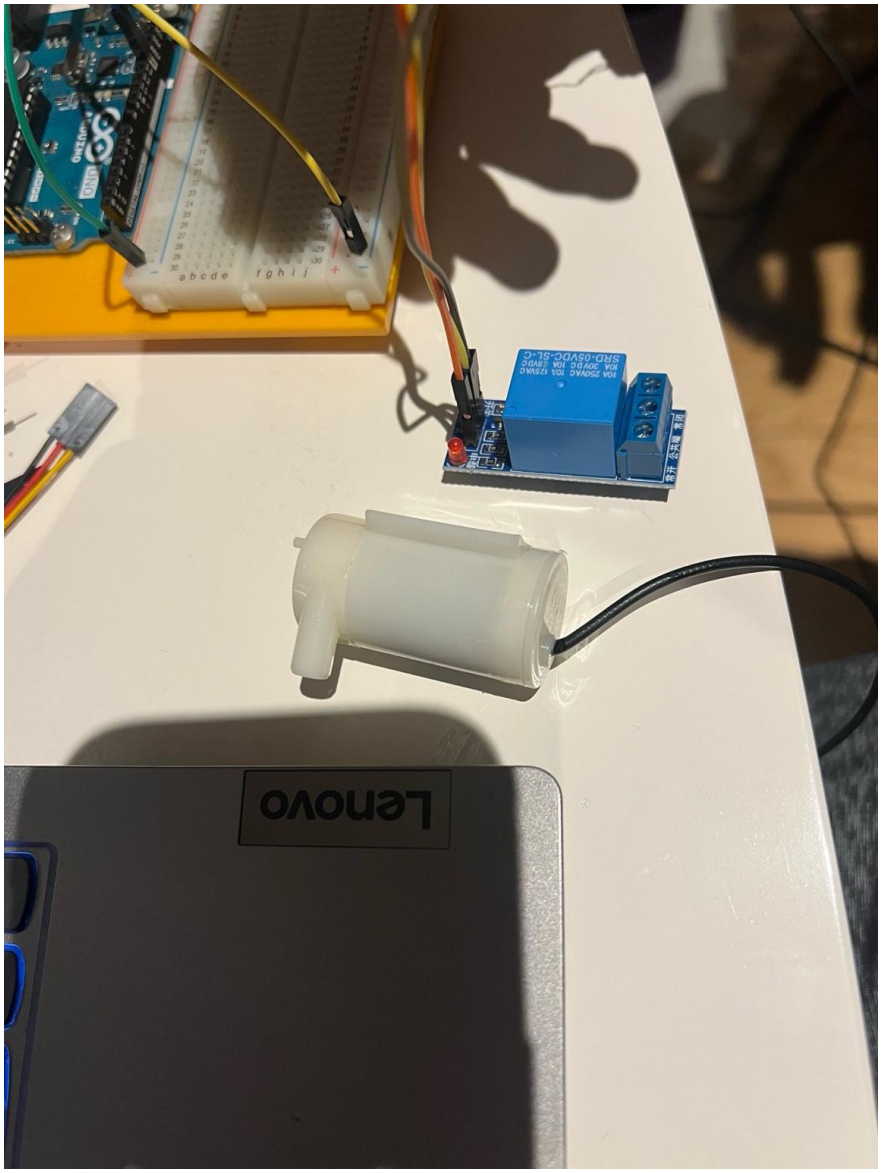

Figure 3: Bench photo – Arduino Uno, relay, and 5–6 V pump.

Figure 4: Elephant enclosure render – body and detachable hat.

Figure 5: Fusion 360 – split parts