

Urban Farming Robot - Variable Reference Guide

IED Project 3 - Arduino UNO with IED Shield

Libraries Required (Must Install Before Upload)

1. LiquidCrystal_I2C (v1.1.2+)

Installation Method:

- Go to <https://www.arduino.cc/reference/en/libraries/liquidcrystal-i2c/>
- Download "LiquidCrystal_I2C-1.1.2.zip"
- Arduino IDE → Sketch → Include Library → Add .ZIP Library
- Select the downloaded ZIP file

2. DHT sensor library (by Adafruit)

Installation Method:

- Arduino IDE → Sketch → Include Library → Manage Libraries
- Search "DHT sensor library"
- Install "DHT sensor library by Adafruit"

3. Adafruit Unified Sensor

Installation Method:

- Arduino IDE → Sketch → Include Library → Manage Libraries
- Search "Adafruit Unified Sensor"
- Install (required dependency for DHT library)

Built-in Libraries (No Installation Needed):

- Wire.h - For I2C communication with LCD
-

IED Arduino Shield Pin Assignments

Based on your IED Shield schematic from Project documents:

Component	Shield Connector	Arduino Pin(s)	Type	Notes
Motors (MX1508 Driver)				
Motor A (Left) IN1	-	D9	PWM Out	Speed control
Motor A (Left) IN2	-	D6	PWM Out	Direction
Motor B (Right) IN3	-	D5	PWM Out	Speed control
Motor B (Right) IN4	-	D3	PWM Out	Direction
Ultrasonic Sensor				
HC-SR04 Trigger	P3	D11	Digital Out	Sends pulse
HC-SR04 Echo	P3	D10	Digital In	Receives echo
IR Sensors				
IR Sensor 1 (Left)	P5	A0	Digital In	Line tracking
IR Sensor 2 (Right)	P6	A1	Digital In	Line tracking
LCD Display				
LCD SDA	P12	A4	I2C	Fixed hardware I2C
LCD SCL	P12	A5	I2C	Fixed hardware I2C
Optional Sensors				
DHT11	-	D12	Digital I/O	Temp & humidity
Moisture Sensor	-	A2	Analog In	Soil moisture
LDR Light Sensor	-	A3	Analog In	Light level
PIR Motion	-	D4	Digital In	Motion detection
Slotted Encoder	-	D2	Digital In	Must be D2 or D3
LED Indicator	-	D13	Digital Out	Built-in LED

Important Notes:

- **D2 and D3** are the only interrupt pins on Arduino UNO (use for encoder)
- **A0-A3** can be used as digital pins if declared with `pinMode(A0, OUTPUT/INPUT)`
- **A4 and A5** are used by LCD I2C - don't use for other sensors

- **9V battery switch** must be turned ON after uploading code for sensors/motors to work

GLOBAL VARIABLES

Ultrasonic Sensor Variables

Variable	Type	Purpose	Range/Notes
<code>g_distanceCm</code>	float	Distance to obstacle (cm)	0-400 cm
<code>g_obstacleDetected</code>	bool	Obstacle within 20cm	true/false

Usage:

```
cpp
if (g_obstacleDetected) {
  stopMotors();
  Serial.println("Obstacle ahead!");
}
```

Moisture Sensor Variables

Variable	Type	Purpose	Range/Notes
<code>g_moistureRaw</code>	int	Raw ADC reading	0-1023
<code>g_moisturePercent</code>	float	Soil moisture %	0-100%
<code>g_needsWatering</code>	bool	Moisture < 30%	true/false

Calibration Required:

```
cpp
```

```
// In code, adjust these values after testing:
g_moisturePercent = map(g_moistureRaw, 1023, 300, 0, 100);
//                               ^^^^ ^^
//                               Dry  Wet
// Test in DRY soil -> note value
// Test in WET soil -> note value
// Update these numbers
```

DHT11 Temperature & Humidity Variables

Variable	Type	Purpose	Range/Notes
<code>g_temperature</code>	float	Temperature in °C	0-50°C typical
<code>g_humidity</code>	float	Relative humidity %	20-90% RH
<code>g_dhtReadSuccess</code>	bool	Valid reading flag	true/false

Usage:

```
cpp
if (g_dhtReadSuccess) {
  lcd.print("Temp: ");
  lcd.print(g_temperature);
}
```

Light Sensor (LDR) Variables

Variable	Type	Purpose	Range/Notes
<code>g_lightRaw</code>	int	Raw ADC reading	0-1023
<code>g_lightPercent</code>	float	Light intensity %	0-100%
<code>g_sufficientLight</code>	bool	Light > 40%	true/false

PIR Motion Sensor Variables

Variable	Type	Purpose	Range/Notes
<code>g_motionDetected</code>	bool	Motion detected	true/false
<code>g_lastMotionTime</code>	unsigned long	Last motion timestamp	milliseconds

IR Line Tracking Variables (2 Sensors)

Variable	Type	Purpose	Range/Notes
<code>g_irLeft</code>	bool	Left IR state	false=on line, true=off line
<code>g_irRight</code>	bool	Right IR state	false=on line, true=off line
<code>g_lineDirection</code>	String	Current direction	"FORWARD", "LEFT", "RIGHT", "STOP"
<code>g_onLine</code>	bool	Robot on line	true/false

IR Sensor Logic (from Project3_10):

Both sensors on BLACK line (0,0) → FORWARD
Left on line, Right off (0,1) → Turn LEFT
Left off, Right on line (1,0) → Turn RIGHT
Both off line (1,1) → STOP (line lost)

Usage:

```
cpp
if(g_lineDirection == "LEFT") {
    turnLeft(g_baseSpeed);
} else if (g_lineDirection == "RIGHT") {
    turnRight(g_baseSpeed);
}
```

Encoder Variables (Distance Tracking)

Variable	Type	Purpose	Range/Notes
<code>g_encoderPulses</code>	volatile unsigned long	Total pulses counted	Incremented by ISR
<code>g_distanceTravelled</code>	float	Distance in meters	Calculated from pulses
<code>PULSES_PER_METER</code>	const float	Calibration constant	Must calibrate!

Calibration Steps:

1. Mark a 1-meter straight line
2. Reset encoder: `g_encoderPulses = 0;`
3. Drive robot exactly 1 meter
4. Note the pulse count
5. Update: `const float PULSES_PER_METER = [your_pulse_count];`

Motor Control Variables

Variable	Type	Purpose	Range/Notes
<code>g_baseSpeed</code>	int	Default motor speed	0-255 (PWM)
<code>g_motorsRunning</code>	bool	Motors active	true/false
<code>g_lineFollowMode</code>	bool	Line following enabled	true/false

Motor Speed Guide:

- 0 = Stop
- 50-100 = Slow (good for precise movements)
- 150 = Medium (default)
- 200-255 = Fast

System Timing Variables

Variable	Type	Purpose	Range/Notes
<code>g_lastSensorUpdate</code>	unsigned long	Last sensor read time	milliseconds
<code>g_lastIRUpdate</code>	unsigned long	Last IR read time	milliseconds
<code>SENSOR_INTERVAL</code>	const unsigned long	Sensor read interval	2000ms (2 seconds)
<code>IR_INTERVAL</code>	const unsigned long	IR read interval	50ms (fast response)

🔑 KEY FUNCTIONS

Motor Control Functions (Based on MX1508 Driver)

```
cpp

void controlMotors(int leftSpeed, int rightSpeed); // Control both motors
void moveForward(int speed);                      // Move straight
void turnLeft(int speed);                         // Turn left (left motor slower)
void turnRight(int speed);                       // Turn right (right motor slower)
void stopMotors();                               // Stop all motors
```

Sensor Reading Functions

```
cpp

int getDistance(int echoPin, int trigPin); // Get ultrasonic distance
void readAllSensors();                    // Read all environmental sensors
void readIRSensors();                    // Read IR sensors for line tracking
void processLineFollowing();              // Execute line following logic
```

Display Functions

```
cpp

void updateLCD(); // Update LCD with current data
void printDebugInfo(); // Print to Serial Monitor for debugging
```

How to Use This Code

1. Upload and Test Sensors

```
cpp

void loop() {
  readAllSensors();
  printDebugInfo(); // Watch Serial Monitor
  delay(1000);
}
```

2. Test Motors Individually

```
cpp

void loop() {
  moveForward(150);
  delay(2000);
  stopMotors();
  delay(1000);
  turnLeft(150);
  delay(1000);
  stopMotors();
}
```

3. Test Line Following

```
cpp

// Enable in setup:
g_lineFollowMode = true;

// Code handles it automatically in loop()
```

Calibration Checklist

Before final demo:

- ☐ **Moisture Sensor:** Test in dry and wet soil, update `map()` values
- ☐ **Light Sensor:** Test in dark and bright conditions
- ☐ **IR Sensors:** Adjust distance knobs for reliable black/white detection
- ☐ **Encoder:** Drive 1 meter, count pulses, update `PULSES_PER_METER`
- ☐ **Motor Speed:** Adjust `g_baseSpeed` for smooth movement

- ☐ **Turn Sensitivity:** Adjust the `0.4` multiplier in `turnLeft()`/`turnRight()`
 - ☐ **LCD Address:** Try 0x27 first, if blank try 0x3F
 - ☐ **9V Battery:** Remember to turn ON switch after uploading!
-

Troubleshooting

Problem	Solution
LCD shows nothing	Check 9V switch ON, try address 0x3F instead of 0x27
Motors don't move	Check 9V switch ON, verify motor connections
IR sensors not detecting	Adjust distance knobs with white object
DHT11 returns NaN	Check wiring, wait 2 seconds between reads
Distance always 0cm	Turn ON 9V switch
Encoder not counting	Use D2 or D3 only (interrupt pins)

Project Requirements Checklist

Compulsory Components:

- ☒ Arduino UNO with IED Shield
- ☒ 2WD Robot Car Base with motors
- ☒ LCD Display (16x2 I2C)
- ☒ Ultrasonic Sensor HC-SR04 **OR** 2x IR Sensors
- ☒ At least one other actuator (motors count!)

Optional Components Used:

- ☒ DHT11 Temperature & Humidity
- ☒ Moisture Sensor
- ☒ PIR Motion Sensor
- ☒ LDR Light Sensor
- ☒ Slotted Encoder

Total Components: Meets IED Project 3 requirements ✓

💡 Tips for Success

1. **Test each sensor separately** before combining
 2. **Use Serial Monitor** (Ctrl+Shift+M) for debugging
 3. **Calibrate sensors** in your actual environment
 4. **Start with low motor speeds** to avoid crashes
 5. **Remember 9V switch** - most common issue!
 6. **Save frequently** - save different versions as you progress
-

📖 Reference to Project Documents

- **Motor Control:** Project3_8_2WD Robot motor control board v1.3.pdf
- **LCD Setup:** Project3_6_Arduino+LCD v1.1.pdf
- **IR Sensors:** Project3_10_Arduino+IR Sensors_v3.1.pdf
- **Ultrasonic:** Project3_9_Arduino+Ultrasound v1.1.pdf
- **Shield Pinout:** Project3_5_Intro to Arduino Shield_v2.2.pdf

Good luck with your project! 🚗🌱