```
#include <Servo.h>

Servo trackerServo;

// ---------------- PIN CONFIGURATION ----------------
const byte LDR_LEFT   = A0;
const byte LDR_CENTER = A1;
const byte LDR_RIGHT  = A2;
const byte SERVO_PIN  = 9;

// ---------------- SERVO LIMITS ----------------
// Adjust these to the safe mechanical limits of your setup.
const int SERVO_MIN_US = 1000;
const int SERVO_MAX_US = 2000;

// Starting position
float servoUs = 1500.0;

// ---------------- SENSOR SETTINGS ----------------
// Number of readings averaged for each sensor
const byte SAMPLES = 8;

// Sensor calibration multipliers
float leftGain   = 1.00;
float centerGain = 1.00;
float rightGain  = 1.00;
```

```
// ---------------- TRACKING SETTINGS ----------------
// Smaller = more sensitive.
// Increase only if the servo reacts to electrical noise.
const float MIN_ERROR = 0.001;

// Maximum movement per control cycle
const float MAX_STEP_US = 2.0;

// Tracking gain
const float GAIN = 25.0;

// Control-loop delay
const unsigned long LOOP_DELAY = 10;

// ------------------------------------------------
// Read an LDR several times and average the result
// ------------------------------------------------
float readSensor(byte pin) {

  long total = 0;

  for (byte i = 0; i < SAMPLES; i++) {
    total += analogRead(pin);
  }

  return (float)total / SAMPLES;
}
```

```
// ------------------------------------------------
// Convert raw sensor readings into normalized values
// ------------------------------------------------
float normalize(float value) {

  value = constrain(value, 0.0, 1023.0);

  return value / 1023.0;
}

// ------------------------------------------------
// Write high-resolution servo position
// ------------------------------------------------
void moveServo(float microseconds) {

  microseconds = constrain(
    microseconds,
    SERVO_MIN_US,
    SERVO_MAX_US
  );

  servoUs = microseconds;

  trackerServo.writeMicroseconds((int)servoUs);
}
```

```
// ------------------------------------------------
// SETUP
// ------------------------------------------------
void setup() {

  Serial.begin(115200);

  trackerServo.attach(
    SERVO_PIN,
    SERVO_MIN_US,
    SERVO_MAX_US
  );

  moveServo(servoUs);

  delay(500);
}

// ------------------------------------------------
// MAIN TRACKING LOOP
// ------------------------------------------------
void loop() {

  // -------- READ ALL THREE SENSORS --------

  float leftRaw   = readSensor(LDR_LEFT);
  float centerRaw = readSensor(LDR_CENTER);
```

```
float rightRaw  = readSensor(LDR_RIGHT);

// -------- APPLY CALIBRATION --------

float left   = normalize(leftRaw)   * leftGain;
float center = normalize(centerRaw) * centerGain;
float right  = normalize(rightRaw)  * rightGain;

// -------- TOTAL LIGHT --------

float total = left + center + right;

if (total > 0.001) {

  /*
     Calculate the position of the light.

     LEFT   = -1
     CENTER =  0
     RIGHT  = +1

     Every sensor participates in the calculation.
  */

  float lightPosition =
    (-1.0 * left +
     0.0  * center +
```

```
   1.0  * right) / total;

/*
   Center sensor modifies the tracking response.

   When the center sensor is strong,
   the panel is close to alignment.
*/

float centerRatio = center / total;

/*
   The center sensor gives additional confirmation
   that the panel is approaching the correct position.
*/

float error = lightPosition;

/*
   Extremely small error is ignored only when it is
   below the physical/electrical resolution limit.

   There is intentionally no large deadband.
*/

if (abs(error) > MIN_ERROR) {
```

```
    // Proportional tracking
    float movement = error * GAIN;

    // Prevent sudden large movements
    movement = constrain(
      movement,
      -MAX_STEP_US,
      MAX_STEP_US
    );

    // Move toward the brighter side
    servoUs += movement;

    // Keep within mechanical limits
    servoUs = constrain(
      servoUs,
      SERVO_MIN_US,
      SERVO_MAX_US
    );

    trackerServo.writeMicroseconds(
      (int)servoUs
    );
  }
}

// -------- SERIAL MONITOR --------
```

```
  Serial.print("L=");
  Serial.print(leftRaw, 1);

  Serial.print("  C=");
  Serial.print(centerRaw, 1);

  Serial.print("  R=");
  Serial.print(rightRaw, 1);

  Serial.print("  Servo=");
  Serial.print(servoUs, 1);

  Serial.print("us");

  Serial.println();

  delay(LOOP_DELAY);
}
```