Session 12

180° servomotors are now controlled with the PCA9685 module.

In order to be able to calculate the angles of movement of the arm from the known position of the object. We will use Al-Kashi's theorem (fig1), the position to be reached will correspond to the centre of the object that equal the centre of the arm gripper.

At first, we will simplify the case study to an arm with 2 mobile segments (we ignore the rotating base and the arm-clamp connection), then we will go to 3 segments and so on.

We retrieve the dimensions of the arm:

r1 = the humerus = piece connected to the first servomotor = 165 mm.

r2 = The ulna = piece connected to the second servomotor = 110 mm.

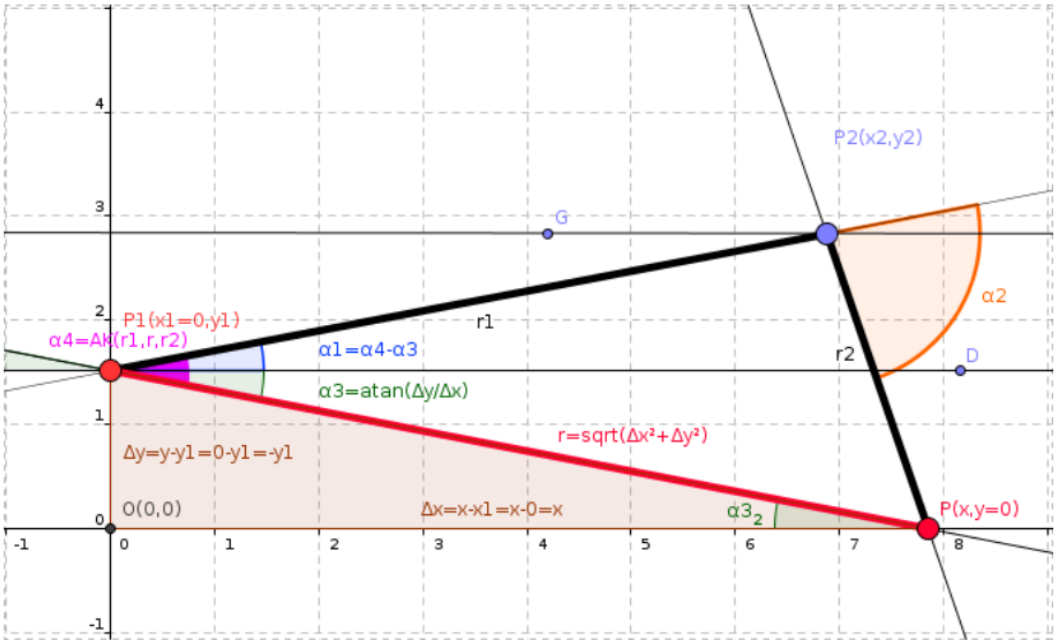

Fig 1: Al Kashi's theorem used to calculate the value of the 2 angles of the 2 servomotors of a 2-axis motorized arm.

The arm is positioned by giving (x, y) coordinate, and via what I understood from the concept of inverse kinematics [1], this is translated to the required servo angles.

The code is composed of two main functions:

1. Movearm (fig 2) responsible for the movement of the arm, we put the coordinates of the point to reach as a parameter then we calculate: angleE = alpha4 (fig1), angleS = alpha2.
2. MoveServo (fig 3) to adjust the measurements, since the Arduino has a margin of error.

The measurements are then transformed from radians to degrees.

```
//  Move the arm to specified x,y position
void moveArm ( float x = 10, float y= 10) {

    // calculate required servo angles using inverse kinematics
    Serial.println("Calculate servo angles:");

    // Work out the length of an imaginery line from the arms shoulder to the x,y position and call it B
    // using pythagoras theorem - length of b squared = x sqared + y squared
    float B = sqrt ( ( x * x ) + ( y * y ) );
    Serial.print("  B = ");     Serial.println( B );

    // Calculate the angle of the imaginary line from the x axis and call it QA
    float QA = atan2 ( y , x );
    Serial.print("  QA = ");     Serial.println( QA );

    // Calculate the angle from the imaginary line to the humerus (using cosine rule) and call it QB
    float B_sq = B * B;    // B squared
    float tvala = hum_sq - uln_sq + B_sq ;
    float tvalb = 2.0 * humerus * B;
    float QB = acos ( tvala / tvalb );
    Serial.print("  QB = ");     Serial.println( QB );

    // Calculate angle of shoulder servo by ading the two calculated angles together
    float angleS = QA + QB;
    Serial.print("  angleS = ");     Serial.println( angleS );

    // Calculate angle of elbow servo
    // this is done using the cosine rule
    tvala = hum_sq + uln_sq - B_sq ;
    tvalb = 2.0 * humerus * ulna;
    float angleE = acos ( tvala / tvalb );
```

Fig 2: first sketch of movearm function

```
// Adjust the mesure
void moveServos ( int s, int e ) {

  // adjust shoulder servo
  s = s * 0.9;

  // adjust elbow servo
  e = e * 0.8;

  // Determine PWM0 pulse width
  pwm0 = s;
  // Write to PCA9685
  pca9685.setPWM(SER0, 0, s);
  // Print to serial monitor
  delay(30);
```

Fig 3: first sketch of moveServo function

[1] https://appliedgo.net/roboticarm/