URO TEST SOAL 3 NOMOR 4

Dynamixel AX-18A merupakan servo yang memiliki torsi yang cukup kuat dan sangat penting untuk pergerakan yang membawa bobot berat. Jelaskan secara bertahap cara kontrol dan komunikasi servo tersebut menggunakan sistem publish subscribe pada Raspberry Pi! (perkirakan juga penggunaan komponen modul lainnya, U2D2 misalnya)

Jawaban:

Sistem *publish-subscribe* pada Raspberry Pi dapat dilakukan dengan berbagai cara, diantaranya adalah dengan MQTT (Message Queue Telemetry Protocol) atau dengan ROS 2. Di bawah ini, hanya ROS 2 yang akan dibahas. Langkah ini beranggapan bahwa seluruh hardware dan libraries yang diperlukan sudah terinstall.

ROS 2:

Pada ROS 2, kita harus membuat 'node' yang memberikan informasi dalam bentuk string messages antar node. Jadi, node pertama yang mempublish data dan node kedua melakukan subscribe ke topik data tersebut sehingga bisa menerima data. Di cara ini, dibutuhkan beberapa komponen, yaitu Raspberry Pi 4, Dynamixel AX-18A Servo, Dynamixel U2D2, dan sebuah power supply.

Secara garis besar, hanya ada dua langkah utama untuk kontrol dan komunikasi Dynamixel AX-18A menggunakan sistem *publish-subscribe*.

1. Membuat publisher di ROS 2 untuk mengirim perintah ke sebuah topik.
2. Membuat subscriber untuk menerima pesan dari topik dan menggerakkan servo menggunakan SDK Dynamixel (sebelumnya sudah harus di install).

Contoh code melakukan publish dan subscribe (bukan contoh code untuk menggerakkan servo, diambil dari dokumentasi ROS jazzy):

1. Membuat package:

   ros2 pkg create --build-type ament_python --license Apache-2.0 py_pubsub
2. Membuat publisher node:
   a. Import

```
import rclpy
from rclpy.node import Node
from std_msgs.msg import String
```

   b. Buat class dengan parameter node

```
class MinimalPublisher(Node):
    def __init__(self):
        super().__init__('minimal_publisher')
        self.publisher_ = self.create_publisher(String, 'topic', 10)
        timer_period = 0.5  # seconds
        self.timer = self.create_timer(timer_period, self.timer_callback)
        self.i = 0

      def timer_callback(self):
```

```
        msg = String()
        msg.data = 'Hello World: %d' % self.i
        self.publisher_.publish(msg)
        self.get_logger().info('Publishing: "%s"' % msg.data)
        self.i += 1
```

c. Main function

```
def main(args=None):
    rclpy.init(args=args)

    minimal_publisher = MinimalPublisher()

    rclpy.spin(minimal_publisher)
    minimal_publisher.destroy_node()
    rclpy.shutdown()

if __name__ == '__main__':
    main()
```

3. Membuat subscriber:

a. Import

```
import rclpy
from rclpy.node import Node
from std_msgs.msg import String
```

b. Buat class dengan parameter node

```
class MinimalSubscriber(Node):
    def __init__(self):
        super().__init__('minimal_subscriber')
        self.subscription = self.create_subscription(
            String,
            'topic',
            self.listener_callback,
            10)
        self.subscription  # prevent unused variable warning

    def listener_callback(self, msg):
        self.get_logger().info('I heard: "%s"' % msg.data)
```

c. Main function

```
def main(args=None):
    rclpy.init(args=args)

    minimal_subscriber = MinimalSubscriber()
    rclpy.spin(minimal_subscriber)
```

```
    # Destroy the node explicitly
    # (optional - otherwise it will be done automatically
    # when the garbage collector destroys the node object)
    minimal_subscriber.destroy_node()
    rclpy.shutdown()

if __name__ == '__main__':
    main()
```

Untuk menggerakkan servo Dynamixel AX-18A, kita harus mengganti message pada publisher menjadi seperti “move_to_position” dan mengubah subscriber untuk memerintahkan servo untuk berpindah.

Sumber: https://docs.ros.org/en/jazzy/Tutorials/Beginner-Client-Libraries/Writing-A-Simple-Py-Publisher-And-Subscriber.html