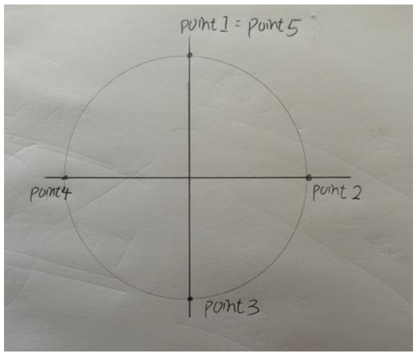
Task1: Manual Robot Setup and Circle Drawing

- Learning Goal:
  - Learn how to properly set up and configure both the hardware (UR10e robot with Robotiq HAND-E gripper) and software (RoboDK offline programming environment) components of a robotic system
  - Learn how to read a robotic support document
  - Program robot arm to draw a circle on a whiteboard
- Tools to Learn:
  - UR10e robot arm with Robotiq HAND-E gripper
  - RoboDK simulator
- **Our Approach:**
  Our approach **utilizes the UR10e's built-in circular movement function**—a default capability that generates half-circular trajectories—to create a complete circle through the sequential execution of two half-circular movements.
  - We designed a path using 5 key points to define two half-circles (see diagram)
  - Added a wait time after the initial touch point to account for whiteboard vibration
  - Utilized unconstrained circle movements to achieve smooth drawing motion



- **Software Implementation:**
  - Set up RoboDK offline programming environment
  - Created a new project with the UR10e robot model and Robotiq HAND-E gripper
  - Defined the whiteboard as a reference plane
  - Configured pressure for gripping marker and movement speed
  - Simulate trajectory and robot movement
- **Hardware Implementation:**
  - Define the entry point (the robot arm will fast reach this point)
  - Define the approach point (the robot arm will slower reach this point to avoid collision)
  - Select Movement P, waypoint: point 1
  - Select wait, time: 0.3s (because when the robot arm touches the whiteboard, the whiteboard vibrates, causing potentially discontinuous lines)
  - Select Circle movement, select unconstrained
  - Define the via point: point 2
  - Define the end point: point 3
  - Define another Circle movement, select unconstrained (starting from point 3, under the same Movement P branch)
  - Define the via point: point 4
  - Define the end point: point 5 (= point 1)

- **Lesson Learned:**
  - How to set up hardware and software for robot simulation and implementation
  - Set an approach point. Approach point creates a controlled deceleration zone before the robot reaches the whiteboard, preventing momentum-related overshooting that could damage the whiteboard or result in inaccurate starting positions
  - Benefit of breaking a complex path into several simpler paths
  - Movement is decided by the reference frame. Setting up a wise reference frame will make the task easier.
- **Demo Results:**
  - All demo videos can be found on GitHub: https://github.com/Kai-Ze612/Intelligent-Machine-Programming-Lab/tree/main/Task1-%20Manual%20Robot%20Setup%20and%20Circle%20Drawing