Manual for turtleleftturn_arc_only() Function

This manual provides a detailed breakdown of the `turtleleftturn_arc_only()` function, used for performing a fixed left turn followed by lane detection in a Gazebo simulation for TurtleBot3 using ROS 2. The function initially performs a left arc based on odometry yaw readings and transitions into lane following by switching to `turtlerightturn.m` once it detects lane lines.

Each section below references the lines of code and explains the operation clearly for easier modification or extension.

1. ROS 2 Node and Topic Initialization (Lines 3-10)

- Creates ROS 2 nodes for general control (`/lanenode`) and odometry (`/odomnode`).
- Subscribes to:
  - Camera image (`/camera/image_raw`)
  - Odometry (`/odom`)
  - Stop signal (`/stop`)
- Publishes to:
  - Velocity command (`/cmd_vel`)
  - Lane-annotated camera output (`/camera/camera_lanes`)

2. Motion and Turn Parameters (Lines 12-21)

- Sobel filter is defined for edge detection.
- `CropValue = 270` crops the image vertically to focus on the road.
- Initializes PID variables (`prevCTE`, `intCTE`) and control timestep `dt = 0.1`.
- Starts the timer and sets `killNode = false` for loop control.
- Defines arc movement:
  - `R = 6.5` sets the radius of the turn.
  - `Lvel = 0.5` defines forward speed.
  - `Avel = Lvel/R` computes required angular velocity.

  - `YawTarget = pi/2 - 0.05` defines how much to turn before stopping (~90 degrees).

3. Starting Orientation and Main Loop (Lines 23-79)

- Line 23: Gets initial yaw using `getYaw()` from `/odom`.

- Line 26-27: Checks for a stop message and exits if needed.

- Line 28-32: Captures and decodes the camera image.

- Lines 33-38: Processes the image:

  - Converts to grayscale

  - Applies yellow mask using HSV

  - Applies Sobel edge filtering and combines yellow and edge gradients

- Lines 39-42: Detects lines using Hough Transform.

- Line 43: Initializes overlay image for visualization.

- Line 46-51: Determines number of lines:

  - If 2+ lines found and time > 10s, switches to `turtlerightturn.m`

  - If 1 line, continue following

  - If no lines, just prints info

4. Arc Turn Control using Yaw (Lines 52-64)

- Uses `getYaw()` again to find current yaw.

- Calculates yaw difference from starting angle (`delta_yaw`).

- If the yaw is within target:

  - Publishes velocity to keep turning

  - Displays current yaw in degrees

- If the yaw exceeds target:

  - Stops the robot

  - Breaks from the loop

- Each loop iteration pauses for `dt` seconds

5. Helper Function: getYaw (Lines 66-69)

- Extracts quaternion from odometry.

- Converts quaternion to Euler angles.

- Returns yaw (rotation around Z-axis).

6. Helper Function: createMask (Lines 70-76)

- Converts image to HSV.

- Applies color filtering to isolate yellow.

- Returns binary mask and a masked RGB image where non-yellow pixels are removed.