DUCKIETOWN

Demo: lane following.

Architecture:

make demo-lane-following

↳ Makefiles/ Makefile demos.mk.

source environment.sh — start ROS env.

↳ source set-ros-master.sh ← set_ros-master local

source set-vehicle-name.sh

roslaunch duckietown_demo lane-following.launch with line-detector-param-file-name:=$*

→ in catkin_ws/src/ 70-convenience-packages/duckietown-demo/launch

→ launch master.launch

→ veh:= "mOOSE"

→ visualization = "true" (def)

→ verbose = "false" (def)

→ param_file_name = ... $* from demo-lane-following or "default" (def) (?)

→ line_detector_param_file_name = "default" (def)

→ anti-instagram = "true" (def)

→ lane_following/stop_line_filter = "false" (def)

IF GROUPS

→ Groups:

✓ Camera : true.

✓ → ~~____~~ cam-info : True

→ camera / raw : false.

✓ wheels → True

odometry → false

✓ Joystick → True

✓ FSM → True

✓ Lane-following → True

✓ → ~~___~~ line-detector → true

✓ → ground_projection → true Lo visualizate : true

✓ → lane-filter → true

→ ~~stop~~ stop_line_filter → false

✓ → lane_controller → true

Obstacle avoidance → false

Vehicle avoidance → false

April Tags → false

✓ LED → True

→ detector → false

→ joystick → false

→ pattern_switch → false

✓ → emitter → true

→ interpreter → false

Odometry learning → false
coordination → false
Navigation → false
Parallel autonomy → false
Global localization → false
SLAM → false

Localization → false

✓ Anti-instagram → true

Lane demo :

Launch files in Master.

⇒ Camera → camera_node.launch. → veh: moose → config: baseline → param_file_name: default
[2] → camera_info_reader_node.launch.

⇒ wheels → inverse_kinematics_node.launch
[2] → wheels_driver_node.launch.

⇒ Joystick → joy_node.launch
[2] → joy_mapper_node.launch.

⇒ FSM → fsm_node.launch → param_file_name: default
[3] → car_cmd_switch_node.launch
→ logic_gate_node.launch.

⇒ lane following → line_detector_node.launch
[6] → ground_projection.launch
→ duckiebot_visualizer.launch
→ lane_filter_node.launch
→ lane_controller_node.launch
→ lane_pose_visualizer_node.launch

LED_sled_emitter_node.launch

Anti_intrusion → anti_intrusion_node.launch

Camera

- camera_node.launch:
  - → pi_camera/camera_node_sequence.py
    - ⇒ "~image/compressed"
    - ⇐ "~framerate_high_switch"
    - ↔ ~set_camera_info

Summary: output images

- camera_info_reader_node.launch:
  - → pi_camera/cam_info_reader_node/default
  - ⇐ camera_node/image/compressed

Summary: ?

Wheels

- inverse_kinematics_node.launch
  - → dogo_ca/inverse_kinematics_node
    - ⇐ "~cmd_car"
    - ⇒ "~wheels_cmd"
    - ↔ ~set_gain
      - ~set limit/baseline/radius/k/lent
      - ~save_calibration

Summary: converts cmd_car to wheel_cmd

- wheels_driver_node.launch
  - → dogo_ca/wheels_driver_node.py
    - ⇒ ~wheels_cmd_executed

Summary: applies wheel_cmd

Summary: Reads the joystick signal

JOYSTICK

- joy_node.launch
  → joy/joy_node
    ⇒ ~~[illegible]~~ joy.

(ROS default)

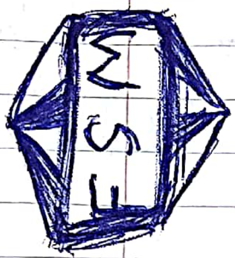
Summary: interprets/map the joystick signal

- joy_mapper_node.launch
  → joy_mapper/joy_mapper_node
    ⇒ ~car_cmd
    ⇒ ~joystick_override
    ⇒ ~parallel_autonomy
    ⇒ anti_instagram_node/click
    ⇒ wheels_driver_node/emergency_stop
    ⇒ ~start_avoidance.
    ⇐ joy

Summary:
⇐ state machine

FSM

- fsm_node.launch
  → fsm/fsm_node.py
    ⇒ ~mode
    ⇔ ~set_state
    ⇒ /moose/camera_node/framerate_high_switch
    ⇒ moose/led_emitter_node/switch
    ⇒ moose/~~[illegible]~~ line_detector_node/switch
    ⇒ moose/lane_filter_node/switch
    ⇐ joy_mapper_node/parallel_autonomy
    ⇐ joy_mapper_node/joystick_override
    ⇐ joy_mapper_node/start_avoidance
    ⇐ simple_coordinator_node/intersection_go
    ⇐ logic_gate_node/intersection_done_and_parallel_autonomy
    ⇐ logic_gate_node/intersection_done_and_not_parallel_autonomy



Summary: decide who sends command.

- car_cmd_switch_node.launch
  - → dagu_car / car_cmd_switch_node
    - ⇒ ~ cmd
    - ⇐ lane_controller_node / car_cmd
    - ⇐ simple_coordinator_node / car_cmd
    - ⇐ joy_mapper_node / car_cmd
    - ⇐ fsm_node / mode
- logic_gate_node.launch
  - → fsm / logicgate_node.py
    - ⇐ joy_mapper_node / parallel_autonomy
    - ⇒ ~ intersection_done_and_parallel_autonomy
    - ⇒ ~ intersection_done_and_not_parallel_autonomy

LANE FOLLOWING

Summary: calls line detector and publishes segment list from camera image.

- line_detector_node.launch
  - → line_detector_node.py
    - ⇒ ~ segment_list (segment list)
    - ⇐ ~ image (image)
    - ⇐ ~ transform (AntiInstagramTransform) (vect 6 float)
    - ⇐ switch (Bool)
    - if verbose
      - ⇒ ~ edge (Image)
      - ⇒ ~ colorSegment (Image)
      - ⇒ ~ image_with_lines (image)

Summary: Projects segments on the ground using camera parameters.

- Ground_projection.launch
  - → ground_projection_node.py
    - ⇒ lineseglist_out (segment list)
    - ⇐ lineseglist_in (segment list)
    - ⇔ ~ estimate_homography (empty & empty)
    - ⇔ ~ get_ground_coordinate (vector2D — point) — point of image → ground coord. % robot
    - ⇔ ~ get_image_coordinate (point — vector 2D) — inverse

21 Juin – 9 juillet
18 juillet – 6 août

- 3 juillet → synthèse brouillon
- 10 juillet → synthèse finale
- 16 juillet → brouillon scénario

lane following

- Duckiebot_visualizer.launch
  - duckiebot_visualizer.py
    - ⇐ ~segment_list (segment list)
    - ⇒ ~segment_list_markers (MarkerArray)

Summary: converts a segment list into MarkerArray.

- lane_filter_node.launch
  - lane_filter_node.py
    - ⇐ ~segment_list (segment_list)
    - ⇐ ~switch (bool)
    - ⇐ ~car_cmd (Twist2D)
    - ⇒ ~lane_pose (LanePose)
    - ⇒ ~belief_img (Image)
    - ⇒ ~ml_img (Image)
    - ⇒ ~entropy (Float)
    - ⇒ ~in_lane (Bool)

Summary: estimates pose / lane + belief from segment lists + dynamic eq (cmd)

double d (lateral offset)
; sigma_d
phi (heading error)
sigma_phi
int status (1: error, 0: normal)
bool in_lane

lane following

- lane_controller_node.launch
  - lane_controller_node
    - ⇐ ~lane_pose (LanePose)
    - ⇒ ~car_cmd (Twist2D)

Summary: uses a PID controller to get back in the center of the lane

- lane_pose_visualizer_node.launch
  - lane_pose_visualizer_node
    - ⇐ ~lane_pose (lane_pose)
    - ⇒ ~lane_pose_markers (MarkerArray)

Summary: converts the lane pose to a MarkerArray

LED

→ led_emitter_node.launch
- led_emitter_node.py
  - ⇐ switch (Bool)
  - ⇐ change_color_pattern (string)
  - ⇒ current_led_state (double)

Summary - Turns LED on

Anti_histogram

→ anti_histogram_node.launch
- anti_histogram_node.py
  - ⇒ corrected_image (Image)
  - ⇒ health (AntiHistogramHealth) (3 doubles)
  - ⇒ transform (AntiHistogramTransform) (4 or 6 doubles)
  - ⇐ uncorrected_image (CompressedImage)
  - ⇐ clicked (Bool)

Summary - Compute transform for the color of image to better all darkness