ME5400A
Robotics Project

Autonomous Mobile Robot for Food Delivery Using Mecanum Wheel Robot

Supervisor: William Lee Wai Leong Co-Supervisor: Marcelo H. Ang
Benjamin Thong Chen Junjie Kent Alvin

Problem

By 2030 almost 1 in 4 Singaporeans will be over 65 years old.

- Prime Minister's Office Singapore [1]

By 2050, 41.5% of Singapore's population will be aged over 60 years old, worse than China.

- United Nations, Economic and Social Commission for Asia and the Pacific. [2]

[1] https://www.pmo.gov.sg/Newsroom/PM-Lee-Hsien-Loong-at-the-Singapore-Ageing-Issues-and-Challenges-Ahead-Book-Launch
[2] https://www.unescap.org/kp/2022/asia-pacific-report-population-ageing-2022-trends-policies-and-good-practices-regarding

By: Benjamin

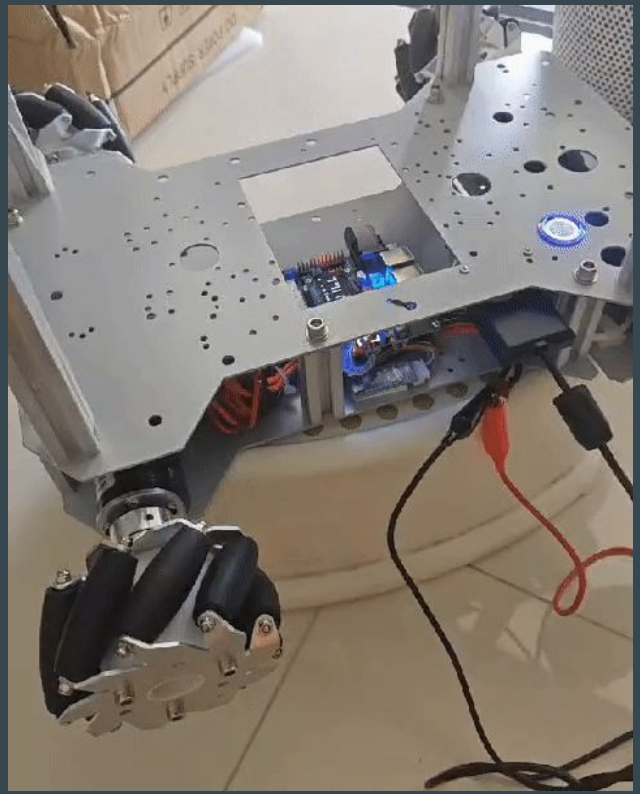
Solution

14th Oct 2023

https://www.youtube.com/watch?v=-f2qKpF7ggE

By: Benjamin

DATA | Hardware



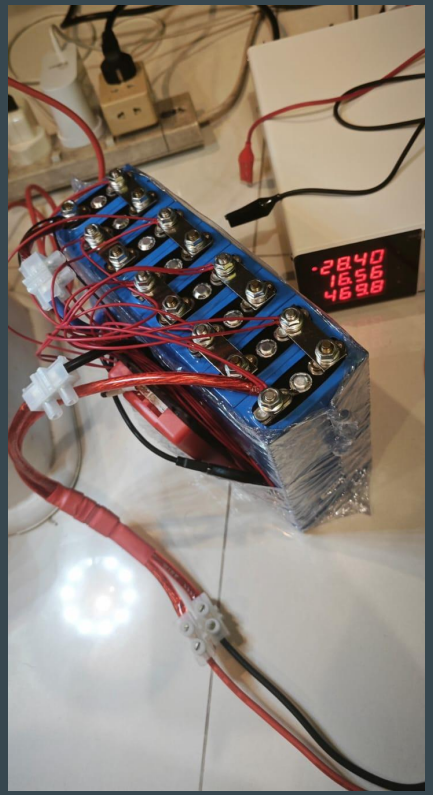
Base configuration

- 40 x 41 cm
- Mecanum 4WD
  - 10 cm wheels
  - 230 rpm
  - 1.2 m/s (Avg walking speed)
  - 13.5 kg.cm
  - 15 kg payload

- No wider than average adult
- Sufficient payload for 2 seniors

By: Benjamin

DATA | Battery DIY



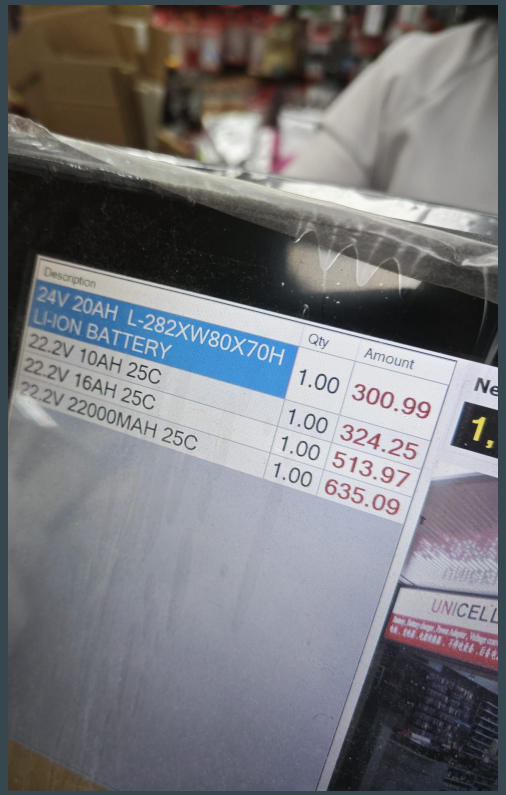
LiFePo4 3.2V Cells. 1C charge. 3C discharge.
8 cells = 25.6V 25.6V 25 Ah = 640 Wh

200W consumption
> 3h continuous operation
1h full recharge

Sufficient for Lunch / Dinner peak hour usage

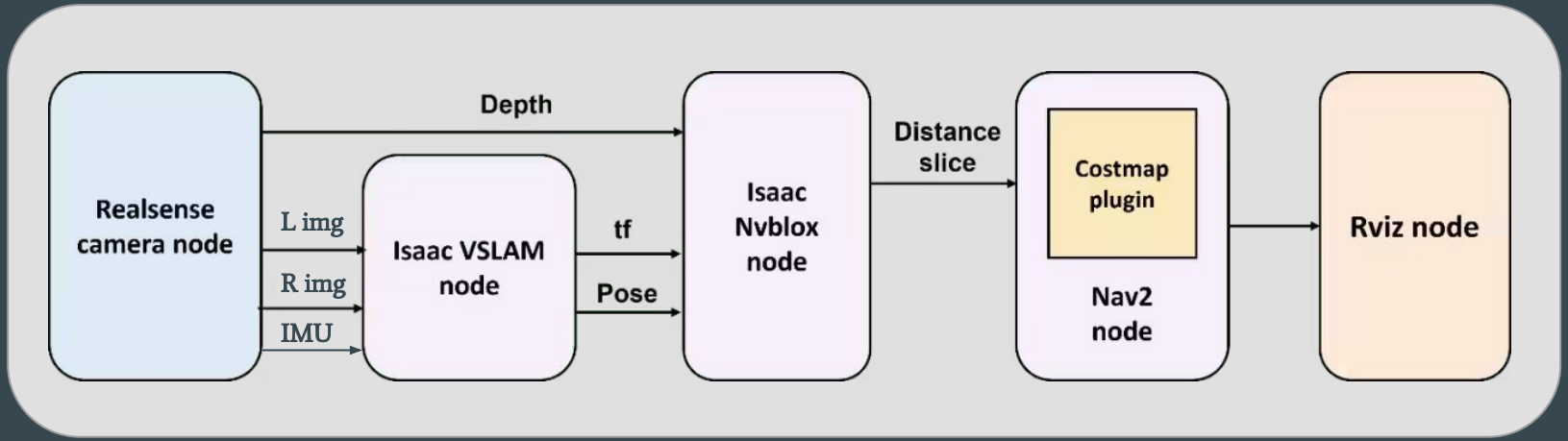
S$230 vs $300 24V 20AH Li-ION 1C

Safer. 10 yrs lifespan. -20 to 60°C range.
100% Depth of Discharge (DoD).



By: Benjamin

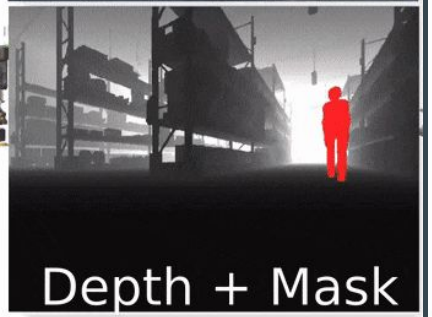
DATA | Vision Pipeline



rc: https://www.arrow.com/en/research-and-events/articles/amr-navigation-using-isaac-ros-vslam-and-nvblox-with-intel-realsense-camera

By: Benjamin

DATA | NvBlox



ROS Topics Subscribed

ROS Topic	Interface	Description
`pointcloud`	sensor_msgs/PointCloud2	Input 3D LIDAR pointcloud. Make sure to set the lidar intrinsic parameters if using this input, as it uses those to convert the pointcloud into a depth image.
`color/image`	sensor_msgs/Image	Optional input color image to be integrated. Must be paired with a `camera_info` message below. Only used to color the mesh.
`color/camera_info`	sensor_msgs/CameraInfo	Optional topic along with the color image above. Contains intrinsics of the color camera.
`depth/image`	sensor_msgs/Image	The input depth image to be integrated. Must be paired with a `camera_info` message below. Supports both floating-point (depth in meters) and `uint16` (depth in millimeters, OpenNI format).
`depth/camera_info`	sensor_msgs/CameraInfo	Required topic along with the depth image. Contains intrinsic calibration parameters of the depth camera.
`transform`	geometry_message/TransformStamped	Odometry as stamped transform messages. Not required if `use_tf_transforms` is set to true.
`pose`	geometry_message/PoseStamped	Odometry as stamped pose messages. Not required if `use_tf_transforms` is set to true.

rc: https://nvidia-isaac-ros.github.io/repositories_and_packages/isaac_ros_nvblox/isaac_ros_nvblox/api/topics_and_services.html

By: Benjamin

DATA | Disparity, Depth, Point Cloud

isaac_ros_ess

960 x 576

ROS Topic	Interface	Description
left/image_rect	sensor_msgs/Image	The left image of a stereo pair.
right/image_rect	sensor_msgs/Image	The right image of a stereo pair.
left/camera_info	sensor_msgs/CameraInfo	The left camera model.
right/camera_info	sensor_msgs/CameraInfo	The right camera model.

ROS Topic	Interface	Description
disparity	stereo_msgs/DisparityImage	The continuous stereo disparity estimation.

isaac_ros_image_proc

Component	Topics Subscribed	Topics Published	Parameters
ImageFormatConverterNode	image_raw, camera_info : The input camera stream	image : The converted image	encoding_desired : Target encoding to convert to.
RectifyNode	image_raw, camera_info : The input camera stream	image_rect, camera_info_rect : The rectified camera stream	output_height : The absolute height to resize to output_width : The absolute width to resize to keep_aspect_ratio : The flag to keep the aspect_ratio when set to true

isaac_ros_visual_slam

ROS Topic	Interface	Description
/stereo_camera/left/image	sensor_msgs/Image	The image from the left eye of the stereo camera in grayscale.
/stereo_camera/left/camera_info	sensor_msgs/CameraInfo	CameraInfo from the left eye of the stereo camera.
/stereo_camera/right/image	sensor_msgs/Image	The image from the right eye of the stereo camera in grayscale.
/stereo_camera/right/camera_info	sensor_msgs/CameraInfo	CameraInfo from the right eye of the stereo camera.
visual_slam/imu	sensor_msgs/Imu	Sensor data from the IMU(optional).

isaac_ros_stereo_image_proc

PointCloudNode	left/image_rect_color : The coloring for the point cloud left/camera_info : The left camera info right/camera_info : The right camera info disparity The disparity between the two cameras	points2 : The output point cloud	use_color : Whether or not the output point cloud should have color. The default value is false. unit_scaling : The amount to scale the xyz points by
DisparityToDepthNode	disparity The disparity image	depth The resultant depth image	N/A

rc:
https://nvidia-isaac-ros.github.io/repositories_and_packages/isaac_ros_dnn_stereo_depth/isaac_ros_ess/index.html#api
https://nvidia-isaac-ros.github.io/repositories_and_packages/isaac_ros_image_pipeline/isaac_ros_image_proc/index.html
https://nvidia-isaac-ros.github.io/repositories_and_packages/isaac_ros_image_pipeline/isaac_ros_stereo_image_proc/index.html#api
https://nvidia-isaac-ros.github.io/repositories_and_packages/isaac_ros_visual_slam/isaac_ros_visual_slam/index.html#api

By: Benjamin

DATA | Disparity, Depth, Point Cloud

isaac_ros_ess

960 x 576

ROS Topic	Interface	Description
left/image_rect	sensor_msgs/Image	The left image of a stereo pair.
right/image_rect	sensor_msgs/Image	The right image of a stereo pair.
left/camera_info	sensor_msgs/CameraInfo	The left camera model.
right/camera_info	sensor_msgs/CameraInfo	The right camera model.

ROS Topic	Interface	Description
disparity	stereo_msgs/DisparityImage	The continuous stereo disparity estimation.

isaac_ros_visual_slam

ROS Topic	Interface	Description
/stereo_camera/left/image	sensor_msgs/Image	The image from the left eye of the stereo camera in grayscale.
/stereo_camera/left/camera_info	sensor_msgs/CameraInfo	CameraInfo from the left eye of the stereo camera.
/stereo_camera/right/image	sensor_msgs/Image	The image from the right eye of the stereo camera in grayscale.
/stereo_camera/right/camera_info	sensor_msgs/CameraInfo	CameraInfo from the right eye of the stereo camera.
visual_slam/imu	sensor_msgs/Imu	Sensor data from the IMU(optional).

isaac_ros_image_proc

Component	Topics Subscribed	Topics Published	Parameters
ImageFormatConverterNode	image_raw , camera_info : The input camera stream	image : The converted image	encoding_desired : Target encoding to convert to.
RectifyNode	image_raw , camera_info : The input camera stream	image_rect , camera_info_rect : The rectified camera stream	output_height : The absolute height to resize to output_width : The absolute width to resize to keep_aspect_ratio : The flag to keep the aspect_ratio when set to true

mono8 -> rgb8

960 x 576

isaac_ros_stereo_image_proc

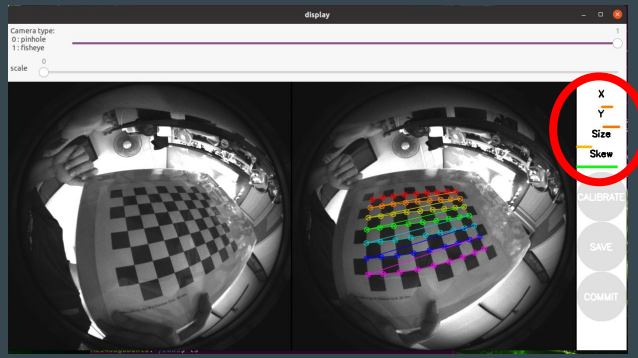
PointCloudNode	left/image_rect_color : The coloring for the point cloud left/camera_info : The left camera info right/camera_info : The right camera info disparity The disparity between the two cameras	points2 : The output point cloud	use_color : Whether or not the output point cloud should have color. The default value is false. unit_scaling : The amount to scale the xyz points by
DisparityToDepthNode	disparity The disparity image	depth The resultant depth image	N/A

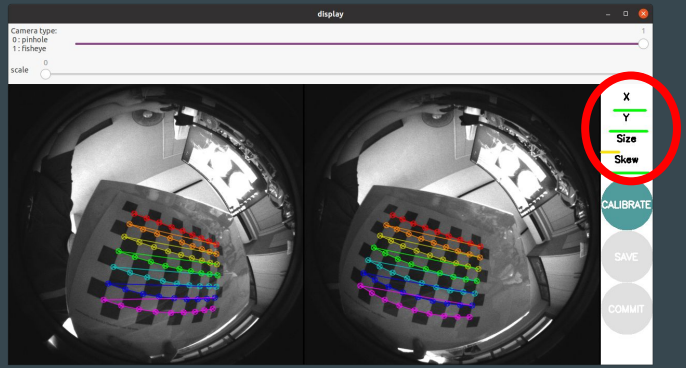
rc:
https://nvidia-isaac-ros.github.io/repositories_and_packages/isaac_ros_dnn_stereo_depth/isaac_ros_ess/index.html#api
https://nvidia-isaac-ros.github.io/repositories_and_packages/isaac_ros_image_pipeline/isaac_ros_image_proc/index.html
https://nvidia-isaac-ros.github.io/repositories_and_packages/isaac_ros_image_pipeline/isaac_ros_stereo_image_proc/index.html#api
https://nvidia-isaac-ros.github.io/repositories_and_packages/isaac_ros_visual_slam/isaac_ros_visual_slam/index.html#api

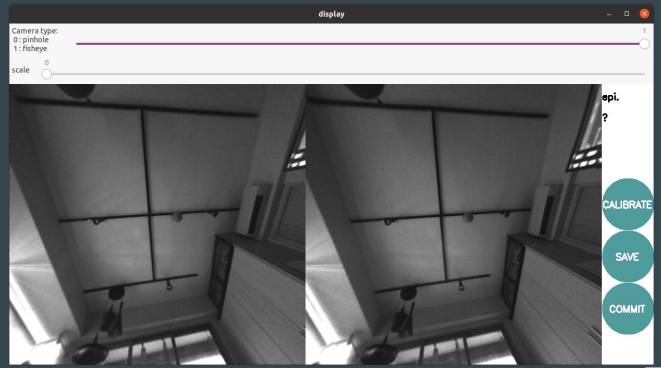
By: Benjamin

DATA | Calibration

Custom ROS2 Python Publisher



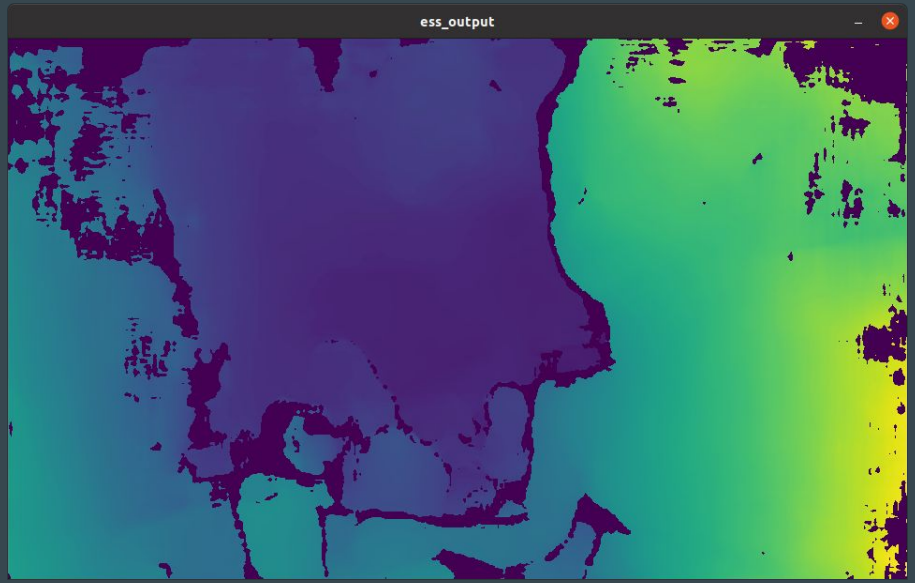


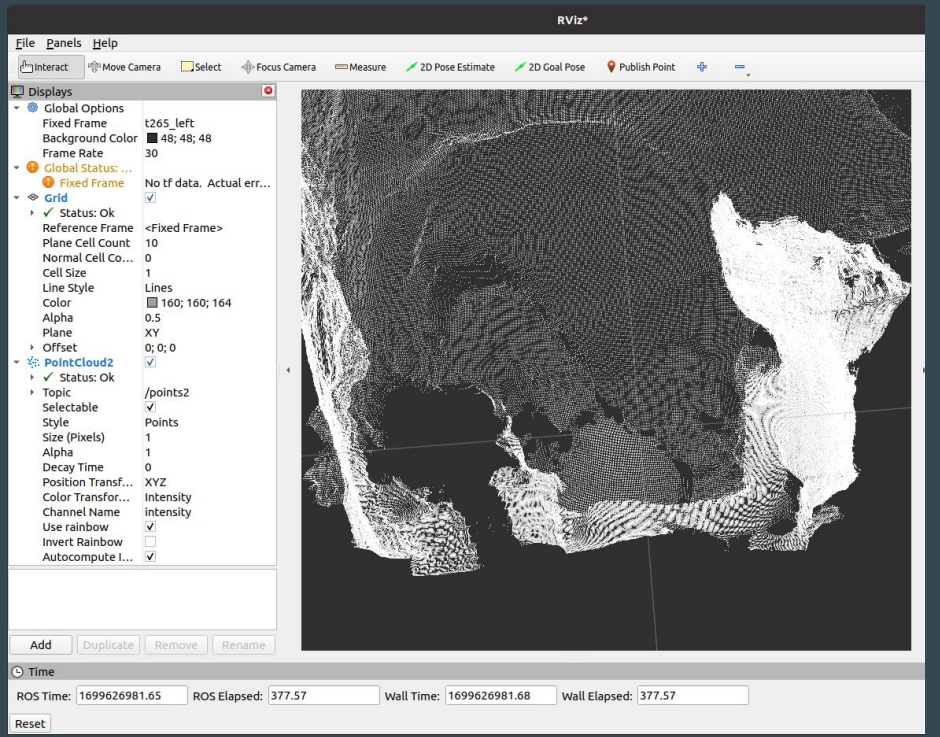


camera_info

By: Benjamin

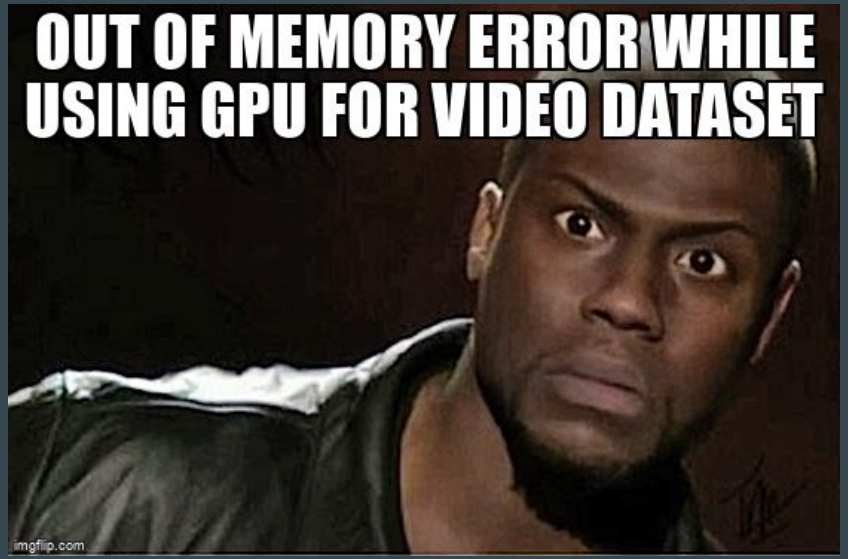
DATA | Disparity, Depth, Point Cloud





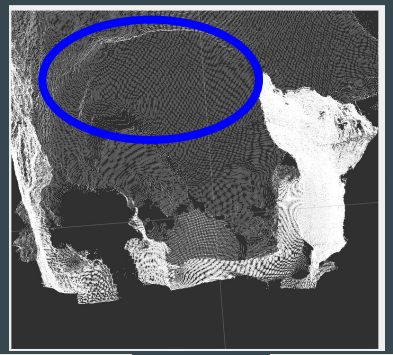
By: Benjamin

DATA | O.M.G.

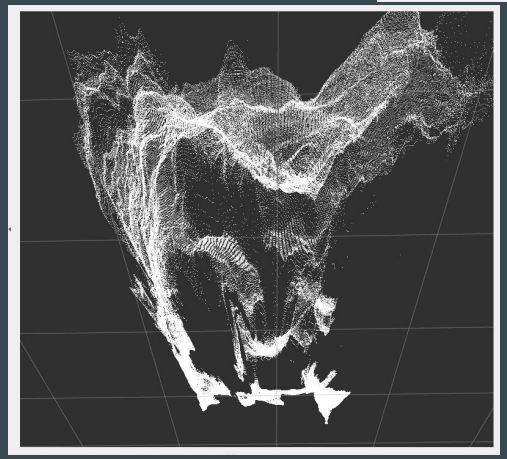


By: Benjamin

DATA | PointCloud to Scan?



Front



45°

Top

Problem in areas with low detail

By: Benjamin

By: Benjamin

DATA | Stereo Visual-Inertial Odometry

isaac_ros_ess

960 x 576

ROS Topic	Interface	Description
left/image_rect	sensor_msgs/Image	The left image of a stereo pair.
right/image_rect	sensor_msgs/Image	The right image of a stereo pair.
left/camera_info	sensor_msgs/CameraInfo	The left camera model.
right/camera_info	sensor_msgs/CameraInfo	The right camera model.

ROS Topic	Interface	Description
disparity	stereo_msgs/DisparityImage	The continuous stereo disparity estimation.

isaac_ros_image_proc

Component	Topics Subscribed	Topics Published	Parameters
ImageFormatConverterNode	image_raw , camera_info : input camera stream	The image	encoding_desired : Target encoding to convert to.
RectifyNode	image_raw , camera_info : input camera stream	rect : ctified camera stream	output_height : The absolute height to resize to output_width : The absolute width to resize to keep_aspect_ratio : The flag to keep the aspect_ratio when set to true

isaac_ros_visual_slam

ROS Topic	Interface	Description
/stereo_camera/left/image	sensor_msgs/Image	The image from the left eye of the stereo camera in grayscale.
/stereo_camera/left/camera_info	sensor_msgs/CameraInfo	CameraInfo from the left eye of the stereo camera.
/stereo_camera/right/image	sensor_msgs/Image	The image from the right eye of the stereo camera in grayscale.
/stereo_camera/right/camera_info	sensor_msgs/CameraInfo	CameraInfo from the right eye of the stereo camera.
visual_slam/imu	sensor_msgs/Imu	Sensor data from the IMU(optional).

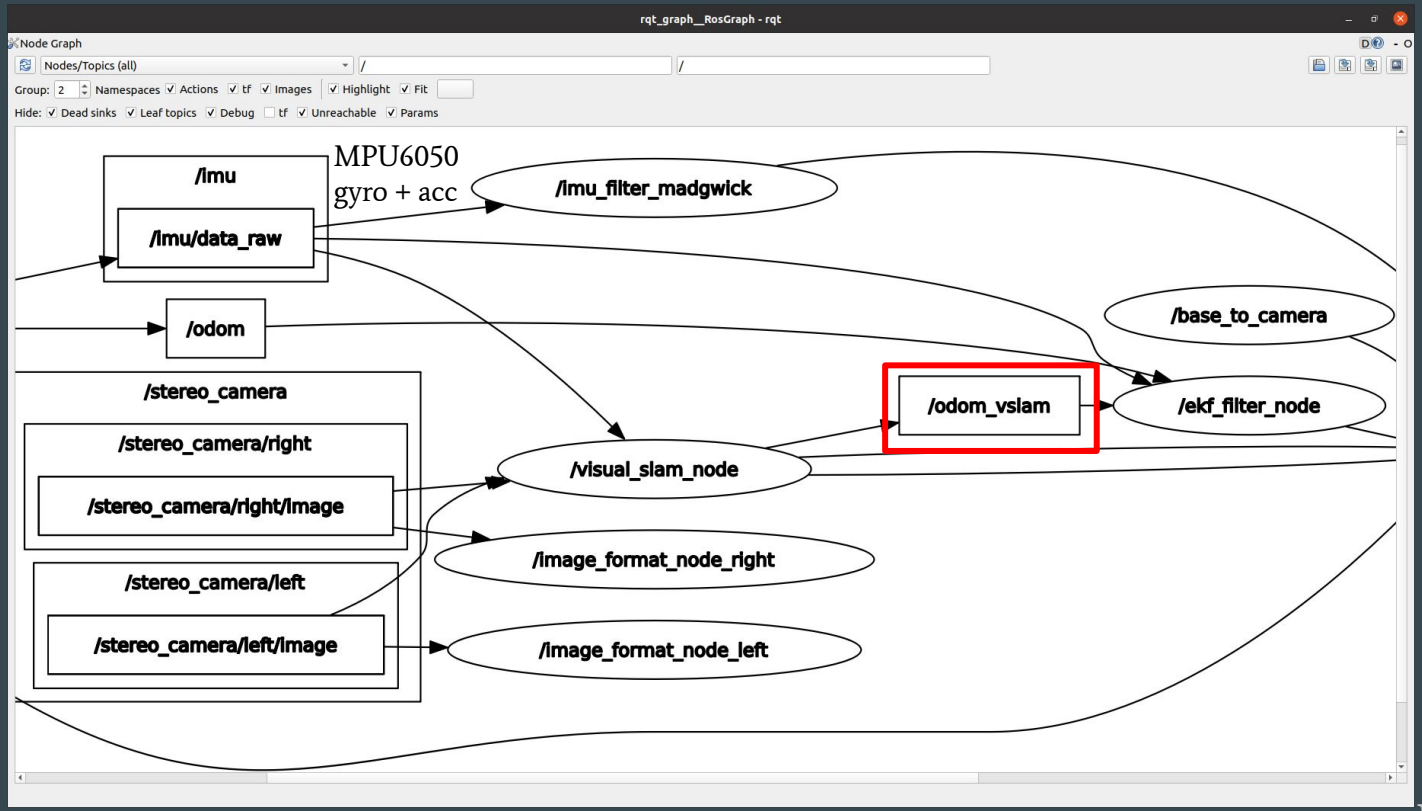
isaac_ros_stereo_image_proc

PointCloudNode	left/image_rect_color : The coloring for the point cloud left/camera_info : left camera info right/camera_ : right camera disparity : disparity between the two cameras	: The output point cloud	use_color : Whether or not the output point cloud should have color. The default value is false. unit_scaling : The amount to scale the xyz points by
DisparityToDepthNode	disparity : The disparity image	The resultant depth image	N/A

rc:
https://nvidia-isaac-ros.github.io/repositories_and_packages/isaac_ros_dnn_stereo_depth/isaac_ros_ess/index.html#api
https://nvidia-isaac-ros.github.io/repositories_and_packages/isaac_ros_image_pipeline/isaac_ros_image_proc/index.html
https://nvidia-isaac-ros.github.io/repositories_and_packages/isaac_ros_image_pipeline/isaac_ros_stereo_image_proc/index.html#api
https://nvidia-isaac-ros.github.io/repositories_and_packages/isaac_ros_visual_slam/isaac_ros_visual_slam/index.html#api

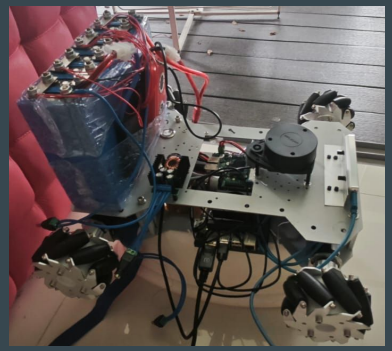
By: Benjamin

DATA | Sensor Fusion

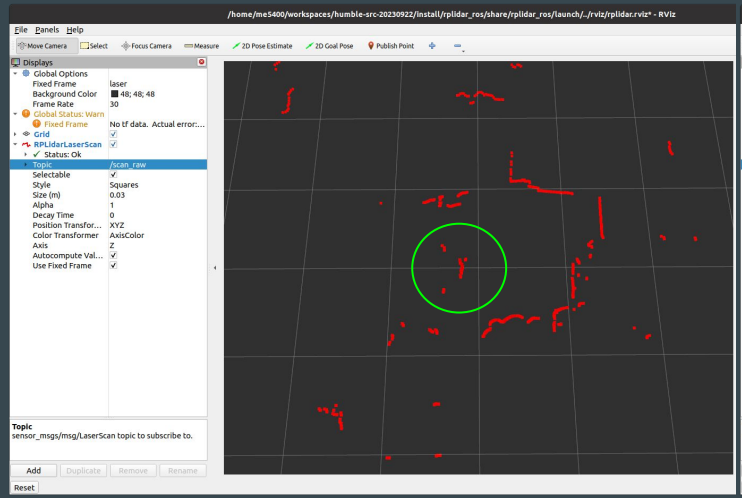


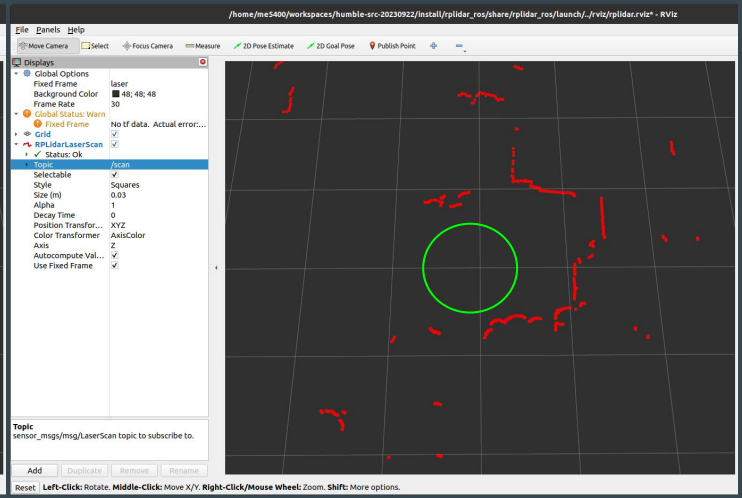
By: Benjamin

DATA | LiDAR Filtering



Custom ROS2 C++ Filter





By: Benjamin

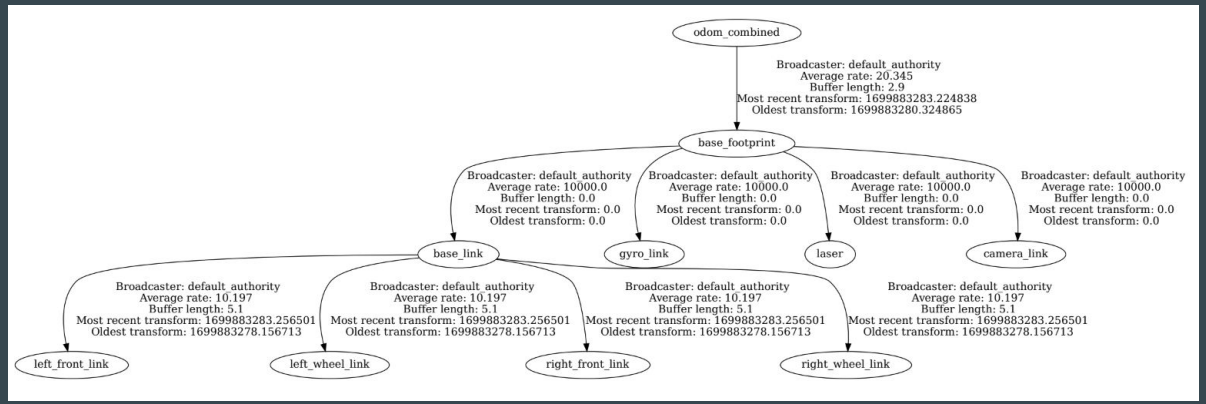
DATA | URDF + Static TF

URDF robot_state_publisher

```xml
<?xml version="1.0" ?>
<robot name="wheeltec_robot">

    <link name="base_link">
        <visual>
            <origin xyz=" 0 0 0.058" rpy="0 0 0" />
            <geometry>
                <box size="0.40 0.24 0.065"/>
            </geometry>
            <material name="white">
                <color rgba="1 1 1 1"/>
            </material>
        </visual>
    </link>

    <joint name="left_wheel_joint" type="continuous">
        <origin xyz="-0.175 0.155 0.05" rpy="1.57 0 0"/>
        <parent link="base_link"/>
        <child link="left_wheel_link"/>
        <axis xyz="0 1 0"/>
    </joint>

    <link name="left_wheel_link">
        <visual>
            <origin xyz="0 0 0" rpy="0 0 0" />
            <geometry>
                <cylinder radius="0.05" length = "0.05"/>
            </geometry>
            <material name="black">
                <color rgba="0 0 0 1"/>
            </material>
        </visual>
    </link>

    <joint name="right_wheel_joint" type="continuous">
        <origin xyz="-0.175 -0.155 0.05" rpy="1.57 0 0"/>
        <parent link="base_link"/>
        <child link="right_wheel_link"/>
        <axis xyz="0 1 0"/>
    </joint>

    <link name="right_wheel_link">
        <visual>
            <origin xyz="0 0 0" rpy="0 0 0" />
            <geometry>
                <cylinder radius="0.05" length = "0.05"/>
            </geometry>
            <material name="black">
                <color rgba="0 0 0 1"/>
```

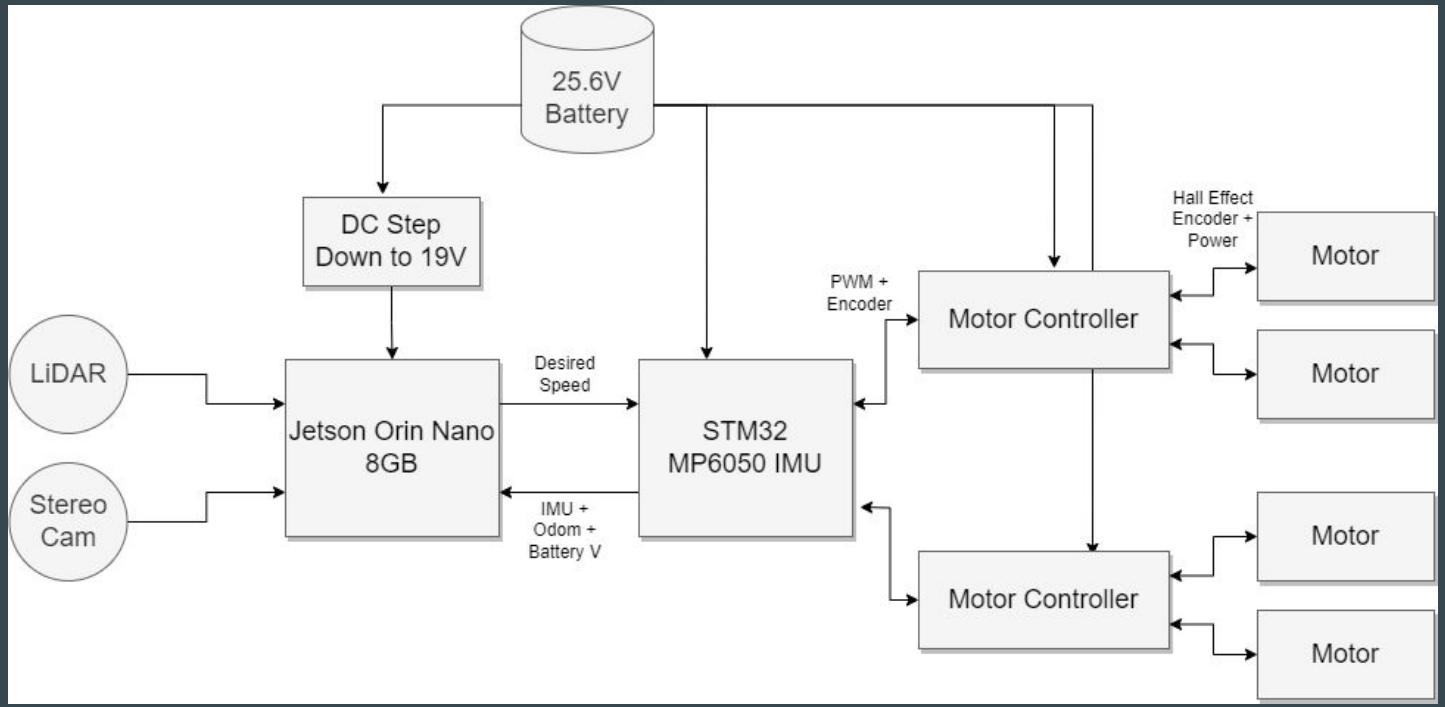
static_transform_publisher

```python
launch_ros.actions.Node(
    package='tf2_ros',
    executable='static_transform_publisher',
    name='base_to_laser',
    arguments=['0.0', '0', '0.12','3.1415', '0','0','base_footprint','laser'],),
launch_ros.actions.Node(
    package='tf2_ros',
    executable='static_transform_publisher',
    name='base_to_camera',
    arguments=['0.0s', '0', '0.20','0', '0','0','base_footprint','camera_link'],),
```



By: Benjamin

DATA | High Level Hardware Diagram



By: Benjamin

DATA | Lessons Learnt

- Difficulty of sourcing for battery in Singapore
- Flashing Jetson SSD without using SD card is a pain
  - Windows instructions didn't work
  - From Ubuntu 22.04 doesn't work - even though ROS Humble meant for 22.04
  - Just use Ubuntu 20.04 !
- Limitations of Jetson Orin Nano 8GB
  - Cannot fit everything into one, even if just for processing single pair of stereo vision
- Noise of stereo vision depth estimation
  - Limits depth map / point cloud generation
- Most resources still in ROS1
  - Porting ROS1 code to ROS2 is not straightforward, especially in C++
- Murphy's Law
  - Many official code repositories / instructions do not work out of the box

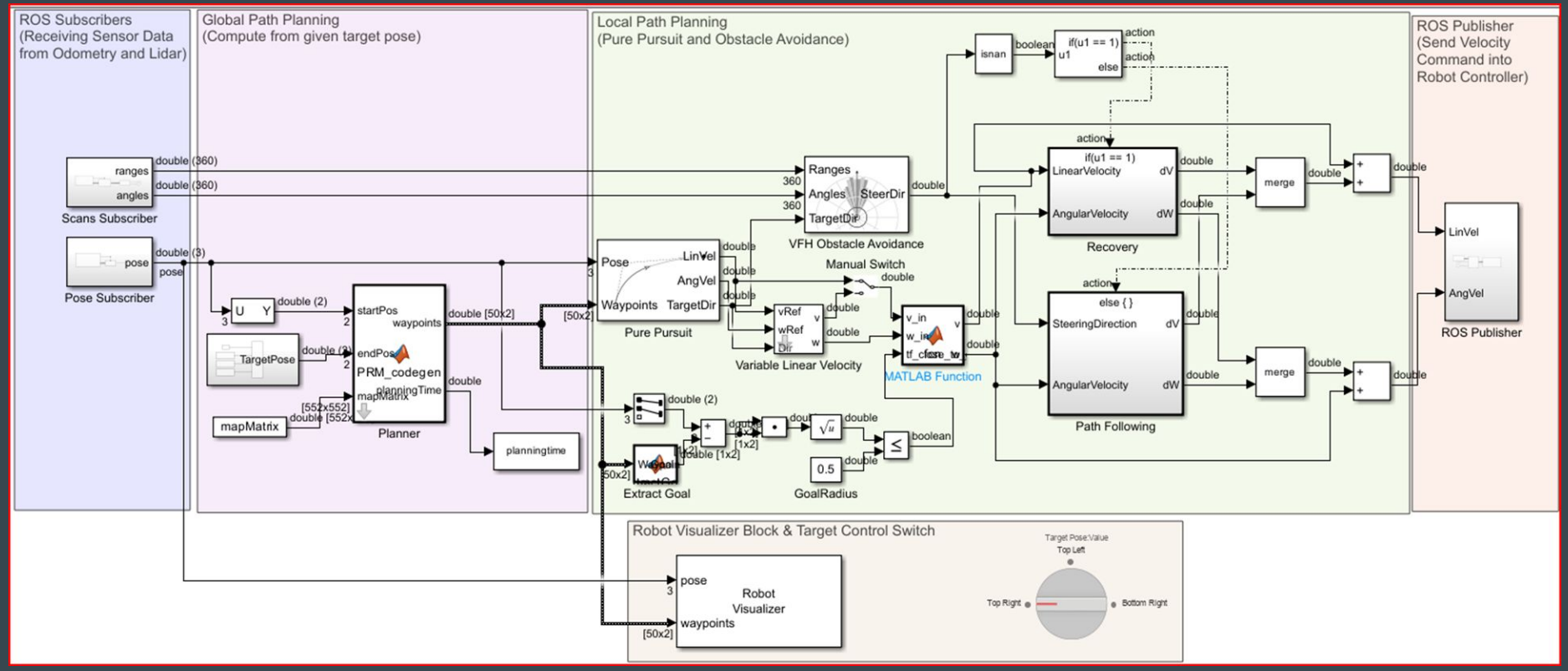
By: Benjamin

COMPUTE | Simulation Step in Computing Part

- Before implementing computation part into real physical robot, first step is do simulation to determine whether the navigation of the robot has a good performance to implement or not
- The navigation including global path planning and local path planning (controller to follow a global path as well as avoiding dynamic obstacle)
- The simulation have been done using Matlab Simulink and ROS2 Nav2 Navigation Stack

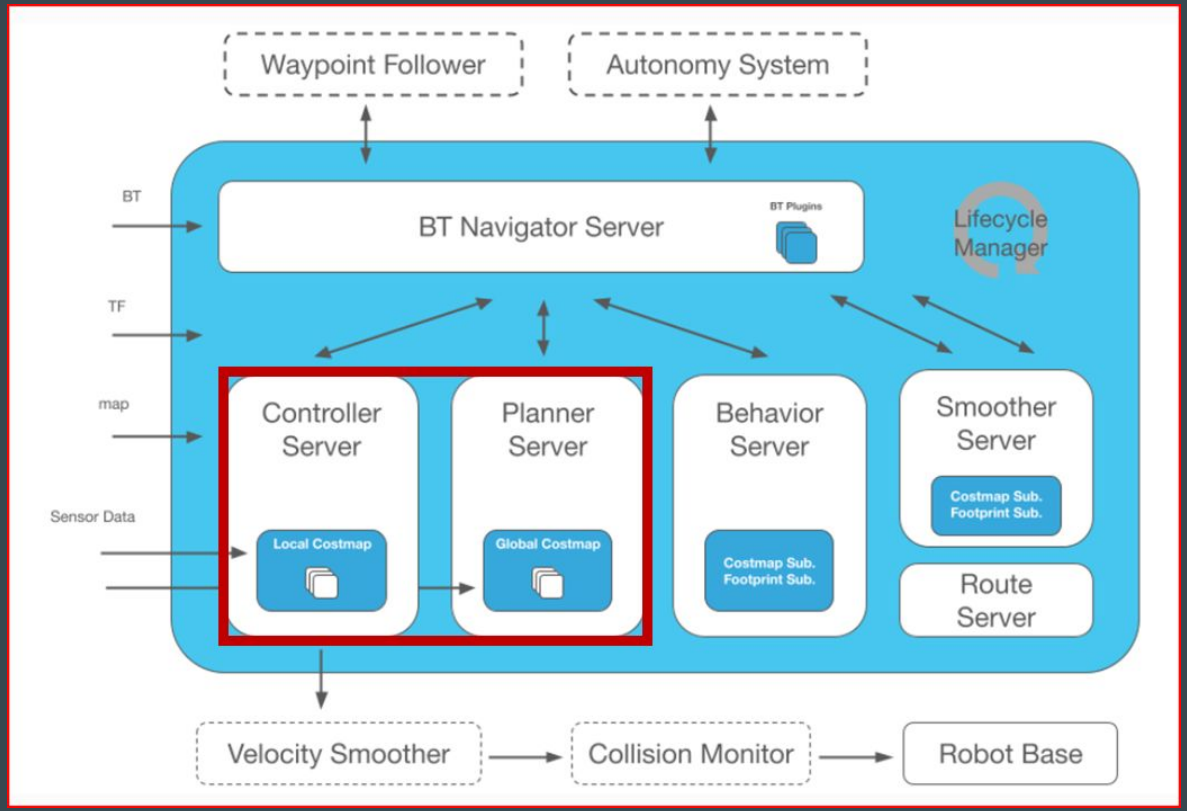
By: Kent

COMPUTE | Computing Path Planning Using Matlab Simulink



By: Kent

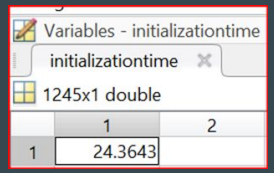
COMPUTE | Nav2 Navigation Stack in ROS2

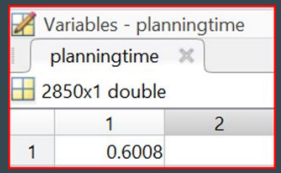


By: Kent

COMPUTE | Comparison Matlab Simulink and ROS2 Nav2

1. Path Planning Time and Initialization Time
   a. In Matlab Simulink, the initialization time take around 24.3643 seconds while path planning time take around 0.6008 seconds.





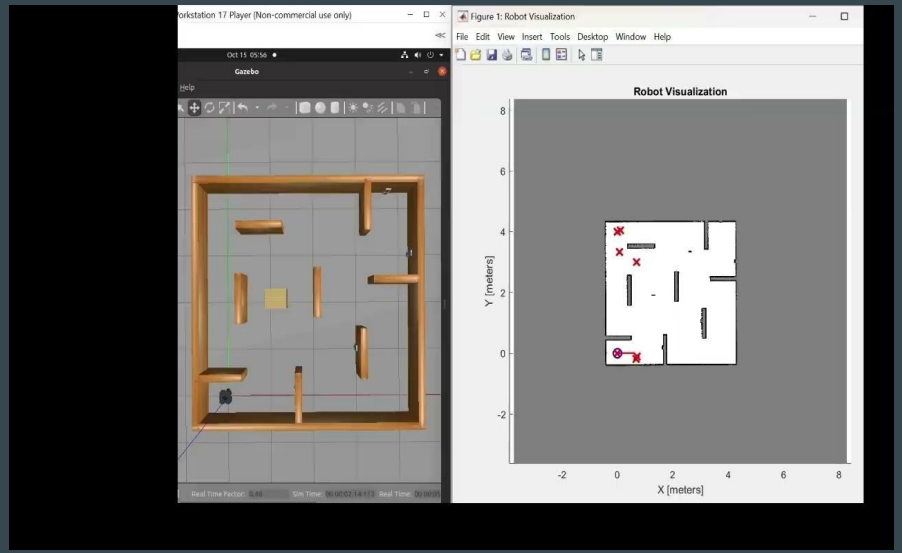
   b. In Nav2, the initialization time take around 5.10 seconds while path planning time take 0.07 seconds.

```
[INFO] [1697501306.399053119] [basic_navigator]: Publishing Initial Pose
[INFO] [1697501311.497638370] [basic_navigator]: Nav2 is ready for use!
Initialization time: 5.10 seconds
[INFO] [1697501311.501081260] [basic_navigator]: Getting path...
[INFO] [1697501311.563487499] [basic_navigator]: Navigating to goal: 11.0 -13.0.
..
Path planning start time: 494648686805 ns
Path planning end time: 494721908645 ns
Path planning time: 0.07 seconds
```

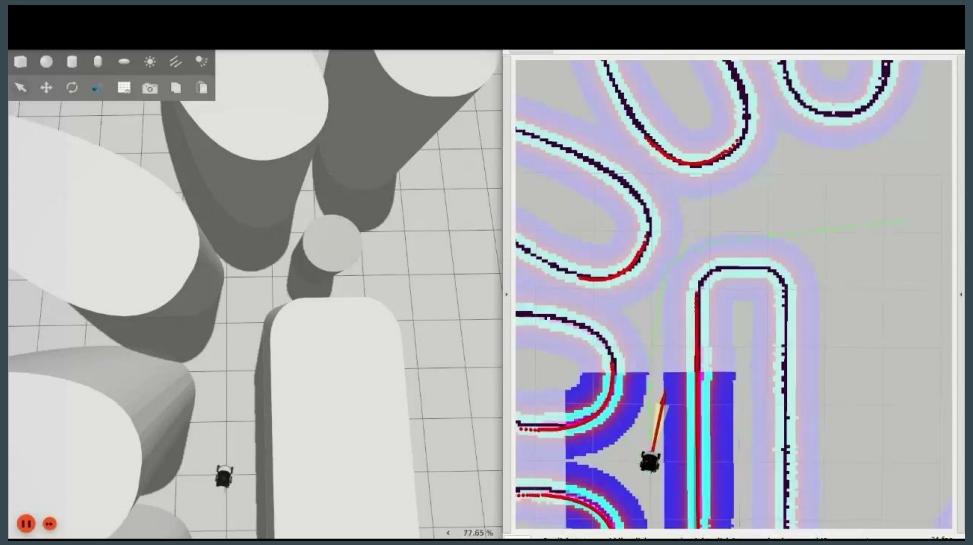
By: Kent

COMPUTE | Comparison Matlab Simulink and ROS2 Nav2

2. Capability to avoid dynamic obstacle



Unable to avoid medium-large dynamic obstacle in Matlab Simulink

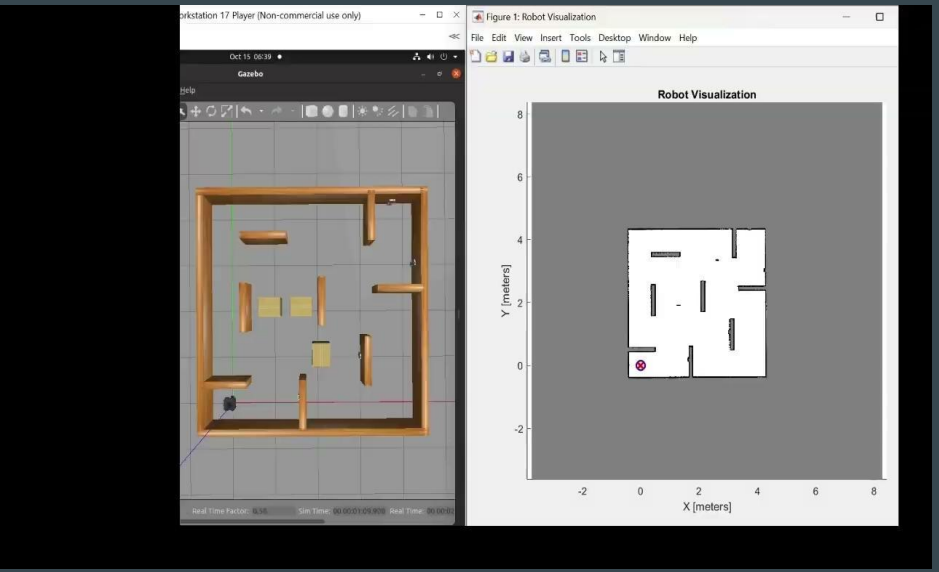


Able to avoid medium-large dynamic obstacle in Nav2

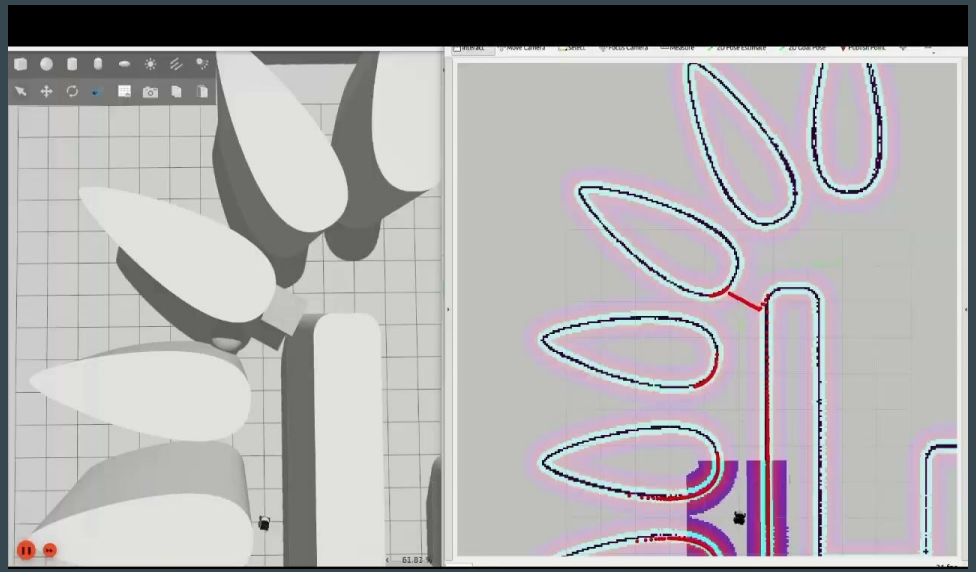
By: Kent

COMPUTE | Comparison Matlab Simulink and ROS2 Nav2

3. Capability to generate a new path when old path get block



Unable to generate a new path when path get block in Matlab Simulink



Able to generate a new path when path get block in Nav2

By: Kent

COMPUTE | Computing Part Step After Simulation

- Find an optimal global path and local path algorithm based on performance metric calculation.
- Implement the chosen global path and local path algorithm using Nav2 Navigation Stack into real physical robot.
- Tuning or adjust some configuration parameter in global and local path planner to provide an optimal behaviour while the robot navigate in its environment.

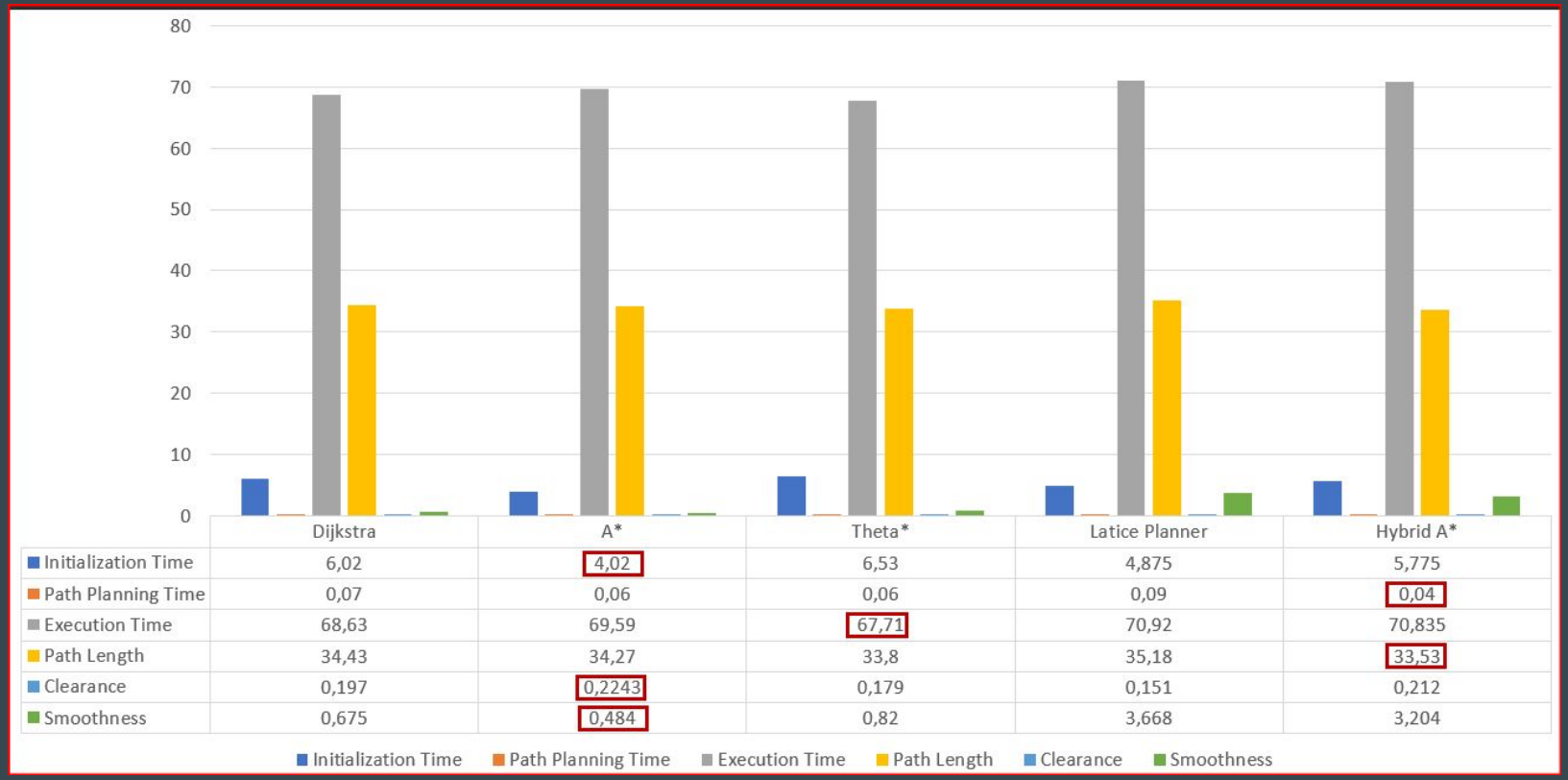
By: Kent

COMPUTE | Performance Metric Calculation

- In path planning, metric calculation used to evaluate the quality and performance of paths generated by path planners.
- Metric that used to evaluate global path planner :
  - Path length
  - Path Planning time
  - Initialization time
  - Execution time
  - Clearance
  - Smoothness
- Metric that used to evaluate local path planner :
  - Tracking Error
  - Clearance
  - Smoothness

By: Kent

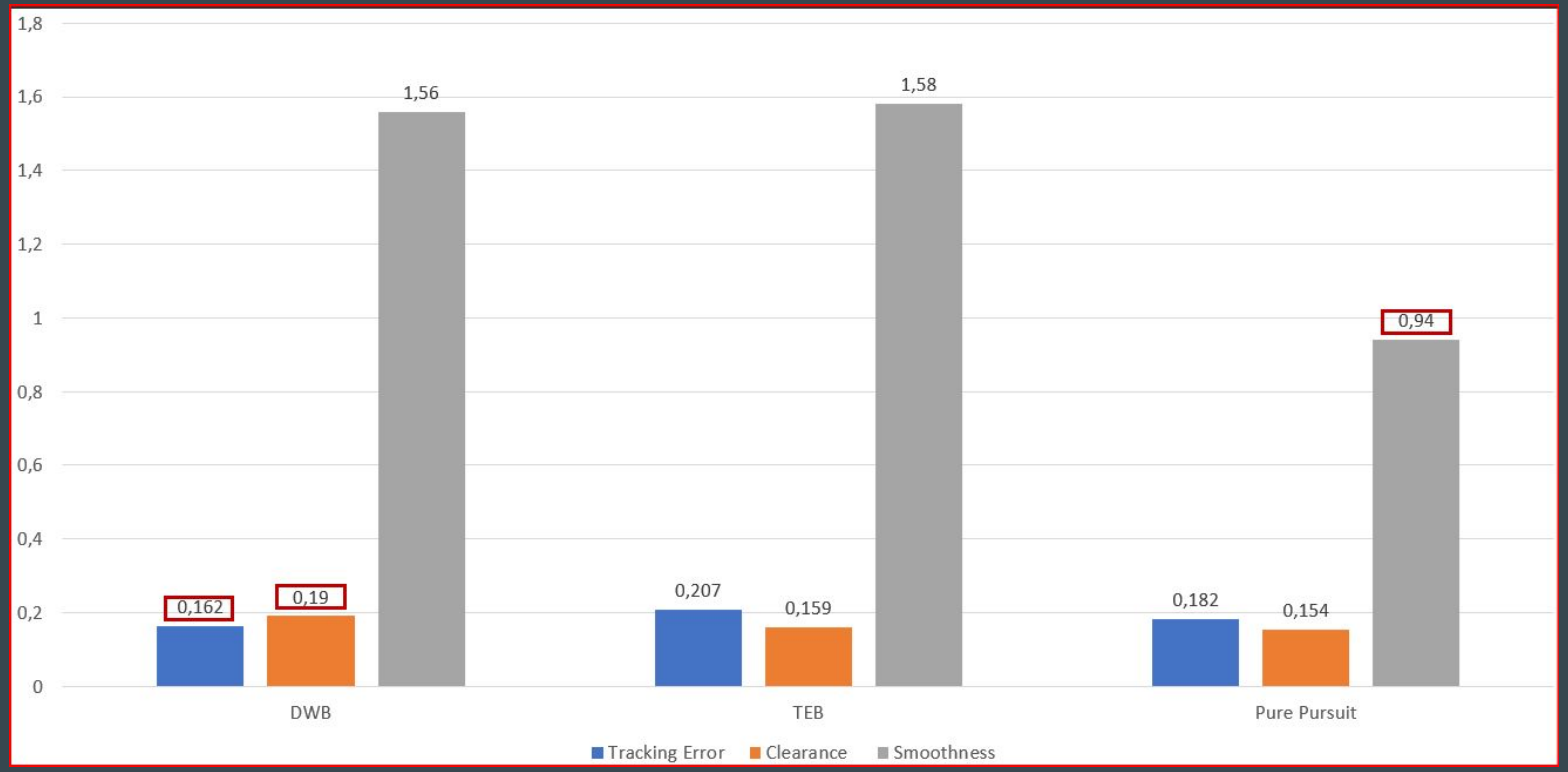
COMPUTE | Overall Metric Calculation for Global Path Planner



	Dijkstra	A*	Theta*	Latice Planner	Hybrid A*
Initialization Time	6,02	4,02	6,53	4,875	5,775
Path Planning Time	0,07	0,06	0,06	0,09	0,04
Execution Time	68,63	69,59	67,71	70,92	70,835
Path Length	34,43	34,27	33,8	35,18	33,53
Clearance	0,197	0,2243	0,179	0,151	0,212
Smoothness	0,675	0,484	0,82	3,668	3,204

By: Kent

COMPUTE | Overall Metric Calculation for Local Path Planner



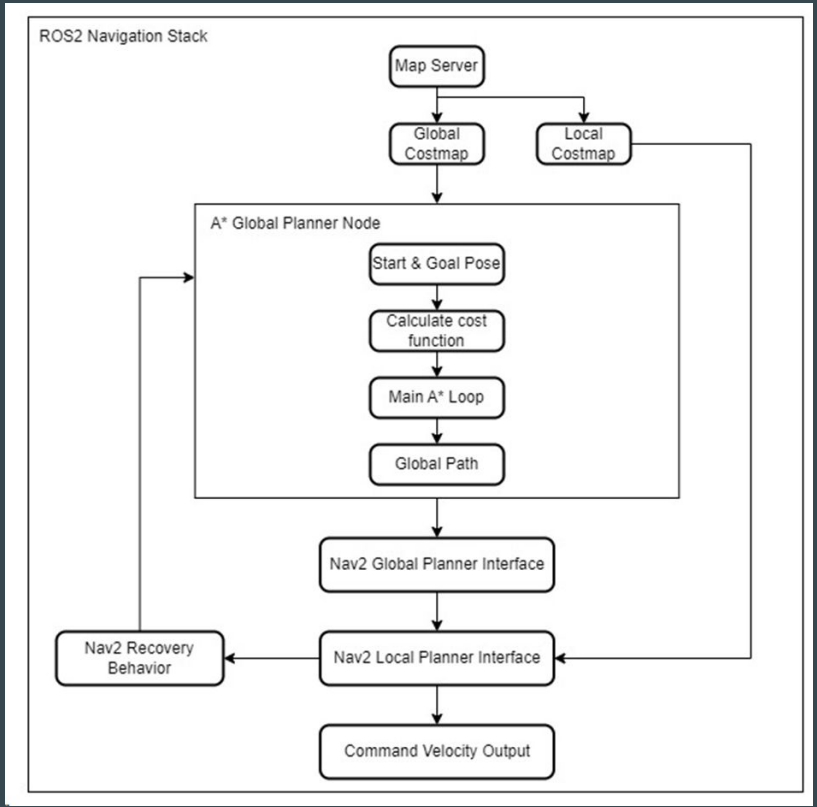
By: Kent

COMPUTE | Implementation Nav2 (A* & DWB) into Real Physical Robot

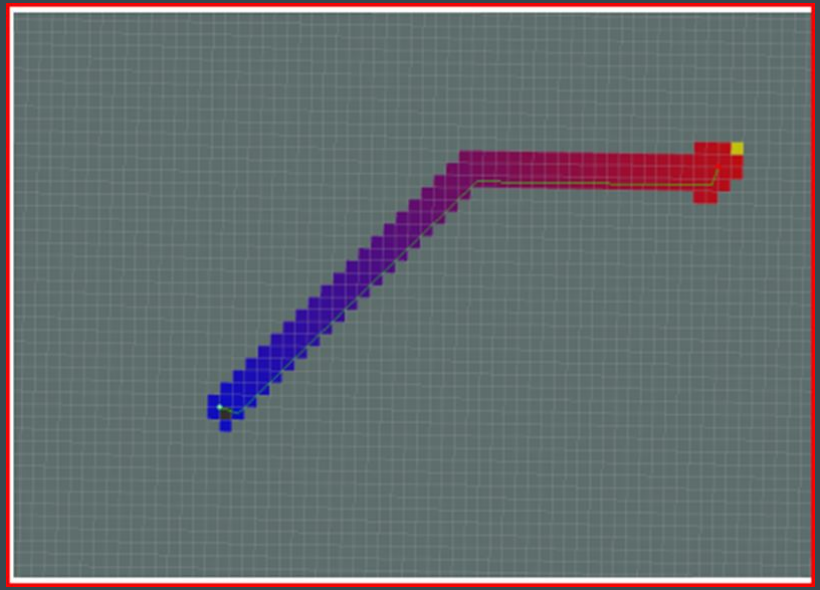
- Perform AMCL used to localize the robot using available sensor data from lidar.
- Set global and local costmap that will be used by planner to generate collision-free path.
- Set and run planner server (global planner) and controller server (local planner) in order for the robot to navigate based on input sensor data and costmap.

By: Kent

COMPUTE | How Global Planner A* Work?



Architecture diagram for global planner



Example how A* planner generate a path to goal position

By: Kent

COMPUTE | Setting Planner Server for Global Path Planner

```yaml
planner_server:
  ros__parameters:
    expected_planner_frequency: 20.0
    use_sim_time: False
    planner_plugins: ["GridBased"]
    GridBased:
      plugin: "nav2_navfn_planner/NavfnPlanner"
      tolerance: 0.5
      use_astar: True
      allow_unknown: True
```

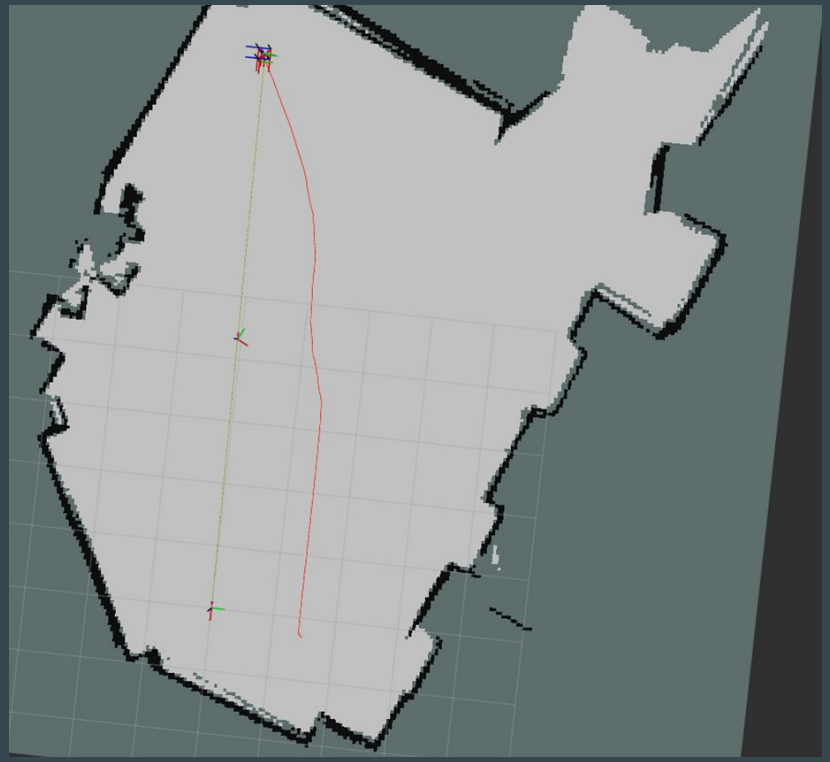
Configuration parameter for planner server (A* Global Planner)

```
while (!openList.empty()) {

    Node current = getLowestFNode(openList);
    openList.erase(std::remove(openList.begin(), openList.end(), current), openList.end());
    closedList.push_back(current);
    if (current == end) {
        return reconstructPath(start, end);
    }
    std::vector<Node> neighbors = getNeighbors(current);
    for (Node neighbor : neighbors) {
        if (std::find(closedList.begin(), closedList.end(), neighbor) != closedList.end()) {
        continue;
        }
        double tentativeG = current.g + distance(current, neighbor);
        if (std::find(openList.begin(), openList.end(), neighbor) == openList.end() || tentativeG < neighbor.g)
{
        neighbor.parent = &current;
        neighbor.g = tentativeG;
        neighbor.f = neighbor.g + heuristic(neighbor, end);
        if (std::find(openList.begin(), openList.end(), neighbor) == openList.end()) {
                openList.push_back(neighbor);
```

Snippet main A* code uses as plugin in Nav2

By: Kent

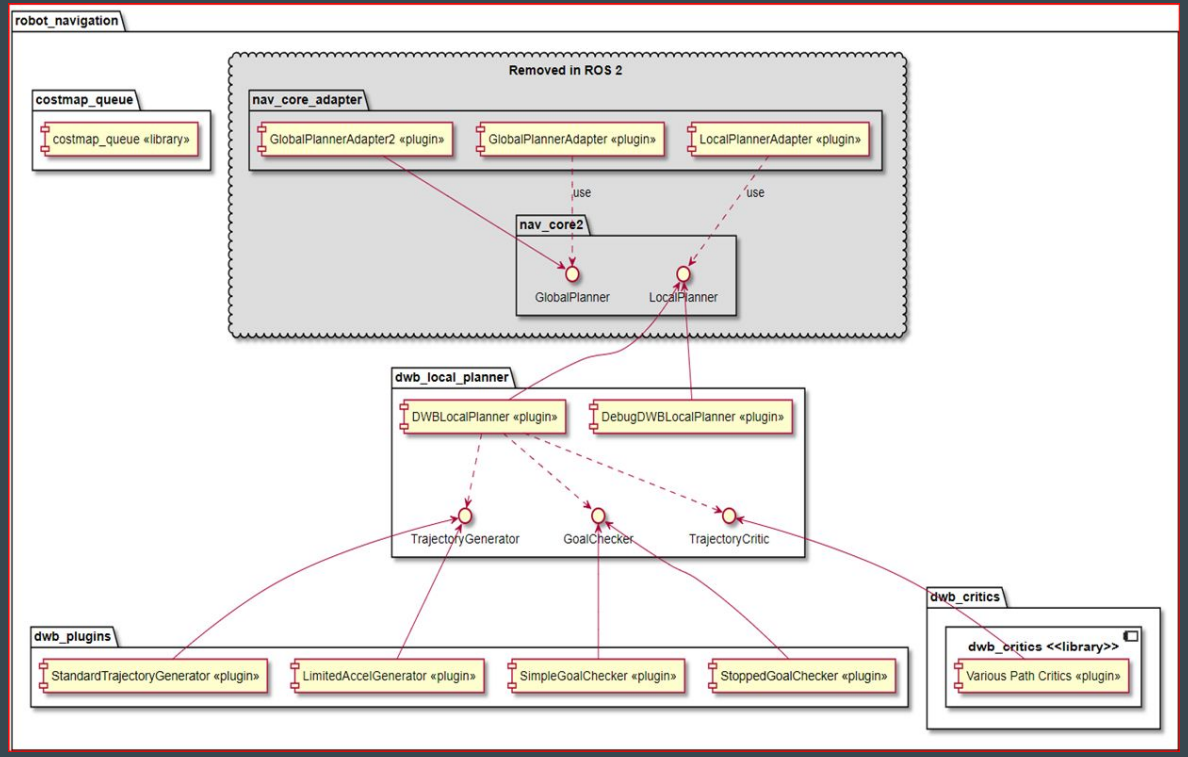
COMPUTE | Setting Planner Server for Global Path Planner



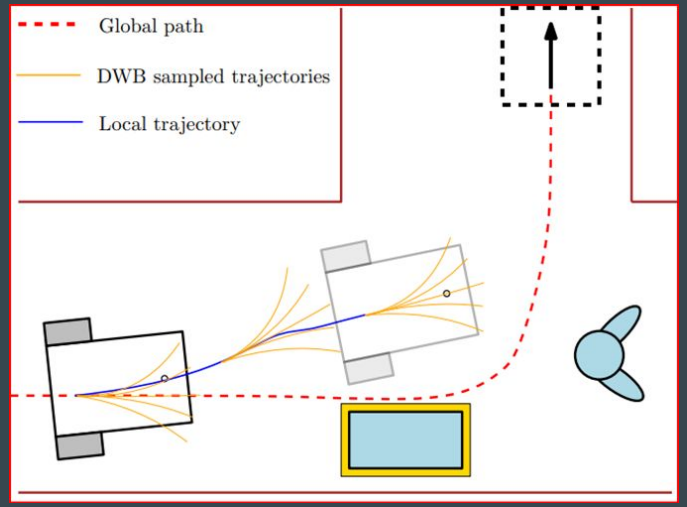
Global path generated from A* Algorithm

By: Kent

COMPUTE | How DWB Local Planner Work?



Architecture Diagram for DWB Local Planner



How DWB generate trajectory and select the best one

By: Kent

COMPUTE | Setting Controller Server for Local Path Planner

```yaml
controller_server:
  ros__parameters:
    min_x_velocity_threshold: 0.001
    min_y_velocity_threshold: 0.001
    min_theta_velocity_threshold: 0.001
    controller_plugins: ["FollowPath"]
    FollowPath:
      plugin: "dwb_core::DWBLocalPlanner"
      min_vel_x: 0.0
      min_vel_y: -0.2
      max_vel_x: 0.5
      max_vel_y: 0.2
      max_vel_theta: 1.5
      min_speed_theta: 0.0
      acc_lim_x: 0.5
      acc_lim_y: 0.0
      acc_lim_theta: 1.0
      decel_lim_x: -0.5
      decel_lim_y: 0.0
      decel_lim_theta: -1.0
      vx_samples: 20
      vy_samples: 20
      vtheta_samples: 20
      critics: ["RotateToGoal", "BaseObstacle", "GoalAlign", "PathAlign", "PathDist", "GoalDist"]
      BaseObstacle.scale: 0.04
      PathAlign.scale: 50.0
      PathAlign.forward_point_distance: 0.1
      GoalAlign.scale: 10.0
      GoalAlign.forward_point_distance: 0.1
      PathDist.scale: 50.0
      GoalDist.scale: 10.0
      RotateToGoal.scale: 32.0
```

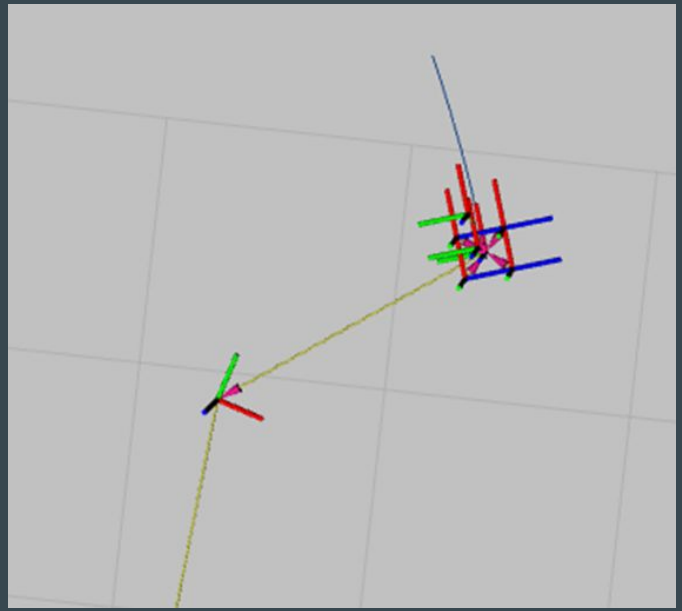
Configuration parameter for controller server (DWB local planner)

```
geometry_msgs::msg::TwistStamped
DWBLocalPlanner::computeVelocityCommands(
  const geometry_msgs::msg::PoseStamped & pose,
  const geometry_msgs::msg::Twist & velocity,
  nav2_core::GoalChecker * /*goal_checker*/)
{
  std::shared_ptr<dwb_msgs::msg::LocalPlanEvaluation> results = nullptr;
  if (pub_->shouldRecordEvaluation()) {
    results = std::make_shared<dwb_msgs::msg::LocalPlanEvaluation>();
  }

  try {
    nav_2d_msgs::msg::Twist2DStamped cmd_vel2d = computeVelocityCommands(
      nav_2d_utils::poseStampedToPose2D(pose),
      nav_2d_utils::twist3Dto2D(velocity), results);
    pub_->publishEvaluation(results);
    geometry_msgs::msg::TwistStamped cmd_vel;
    cmd_vel.twist = nav_2d_utils::twist2Dto3D(cmd_vel2d.velocity);
    return cmd_vel;
  } catch (const nav2_core::PlannerException & e) {
    pub_->publishEvaluation(results);
    throw;
```

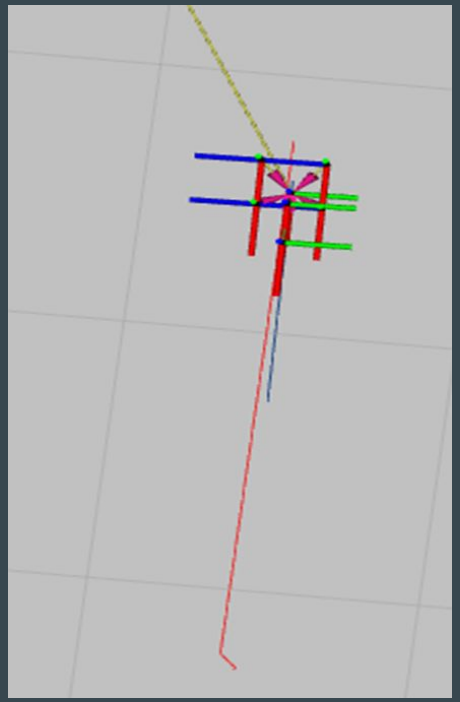
Snippet main/core code to compute output velocity based on DWB algorithm

By: Kent

COMPUTE | Setting Controller Server for Local Path Planner



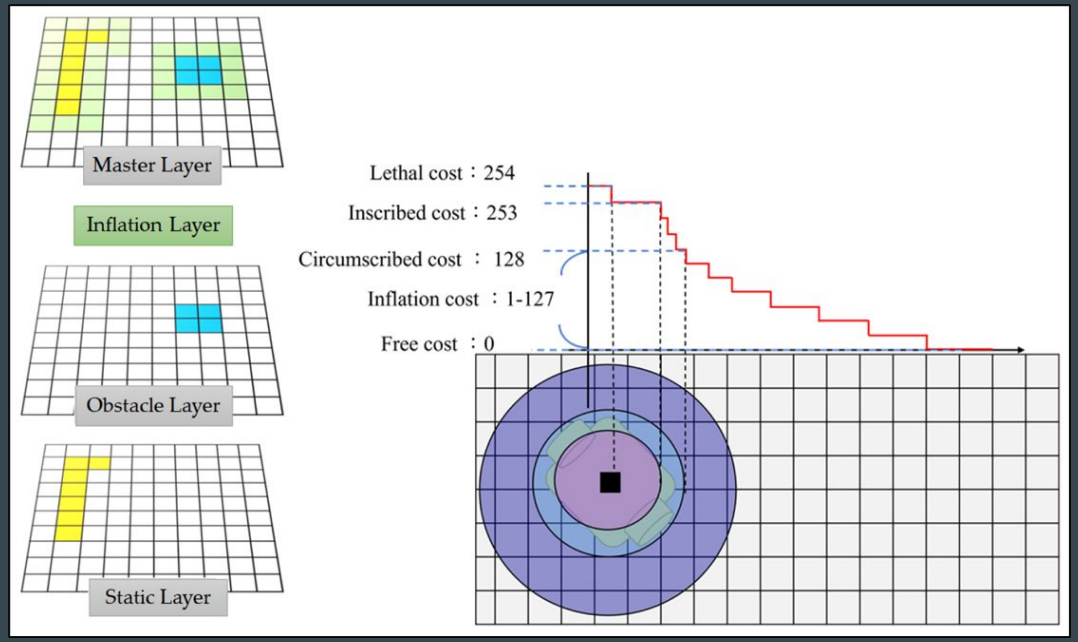
Local path (blue line trajectory) generated by DWB algorithm



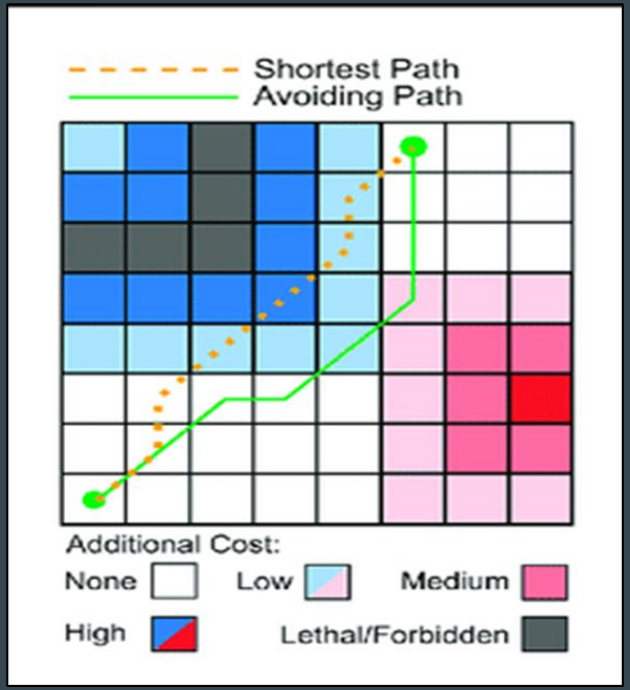
Local path (blue line) following the desired global path (red line)

By: Kent

COMPUTE | How Costmap work?



Difference type of costmap



How costmap used by global planner

By: Kent

COMPUTE | Configure and Setting for Global and Local Costmap

```yaml
global_costmap:
  global_costmap:
    ros__parameters:
      update_frequency: 1.0
      publish_frequency: 1.0
      global_frame: map
      robot_base_frame: base_footprint
      use_sim_time: False
      footprint: "[ [-0.1350, -0.1110], [-0.1350, 0.1110], [0.1350, 0.1110], [0.1350, -0.1110] ]"
      resolution: 0.05
      track_unknown_space: true
      plugins: ["static_layer", "obstacle_layer", "inflation_layer"]
      obstacle_layer:
        plugin: "nav2_costmap_2d::ObstacleLayer"
        enabled: True
        observation_sources: scan
        scan:
          topic: /scan
          max_obstacle_height: 2.0
          clearing: True
          marking: True
          data_type: "LaserScan"
          raytrace_max_range: 10.0
          raytrace_min_range: 0.0
          obstacle_max_range: 2.5
          obstacle_min_range: 0.0
      static_layer:
        plugin: "nav2_costmap_2d::StaticLayer"
        map_subscribe_transient_local: True
      inflation_layer:
        plugin: "nav2_costmap_2d::InflationLayer"
        cost_scaling_factor: 3.0
        inflation_radius: 0.55
      always_send_full_costmap: True
```

Setting parameter for global costmap

```yaml
local_costmap:
  local_costmap:
    ros__parameters:
      update_frequency: 5.0
      publish_frequency: 2.0
      global_frame: odom_combined
      robot_base_frame: base_footprint
      use_sim_time: False
      rolling_window: true
      width: 3
      height: 3
      resolution: 0.05
      footprint: "[ [-0.1350, -0.1110], [-0.1350, 0.1110], [0.1350, 0.1110], [0.1350, -0.1110] ]"
      plugins: ["obstacle_layer", "inflation_layer"]
      inflation_layer:
        plugin: "nav2_costmap_2d::InflationLayer"
        cost_scaling_factor: 3.0
        inflation_radius: 0.25
      obstacle_layer:
        plugin: "nav2_costmap_2d::ObstacleLayer"
        enabled: True
        observation_sources: scan
        scan:
          topic: /scan
          max_obstacle_height: 2.0
          clearing: True
          marking: True
          data_type: "LaserScan"
          raytrace_max_range: 10.0
          raytrace_min_range: 0.0
          obstacle_max_range: 2.5
          obstacle_min_range: 0.0
```

Setting parameter for local costmap

By: Kent

COMPUTE | Configure and Setting for Global and Local Costmap

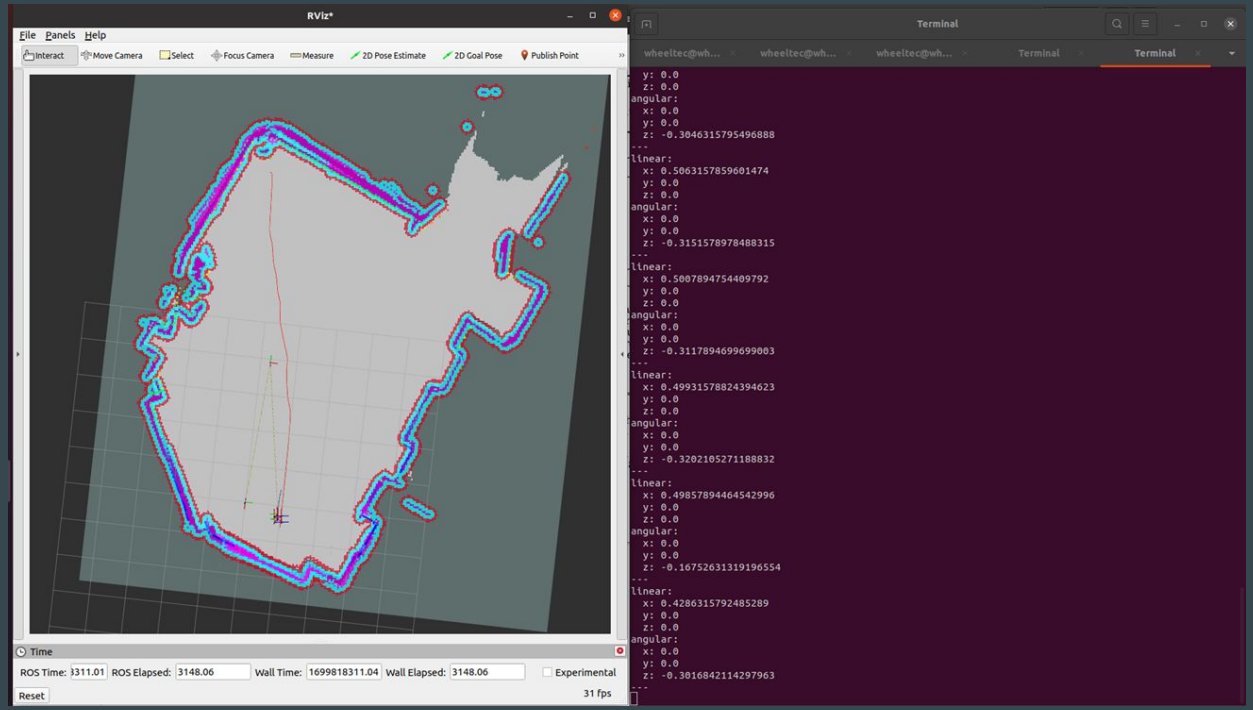
Global Costmap of the current map

Local Costmap of the current map

By: Kent

COMPUTE | Result of Nav2 Navigation Stack Implementation into Robot

- RVIZ showing a generate path while give goal pose command.
- Showing output command velocity in terminal that show the robot move according to this command velocity.



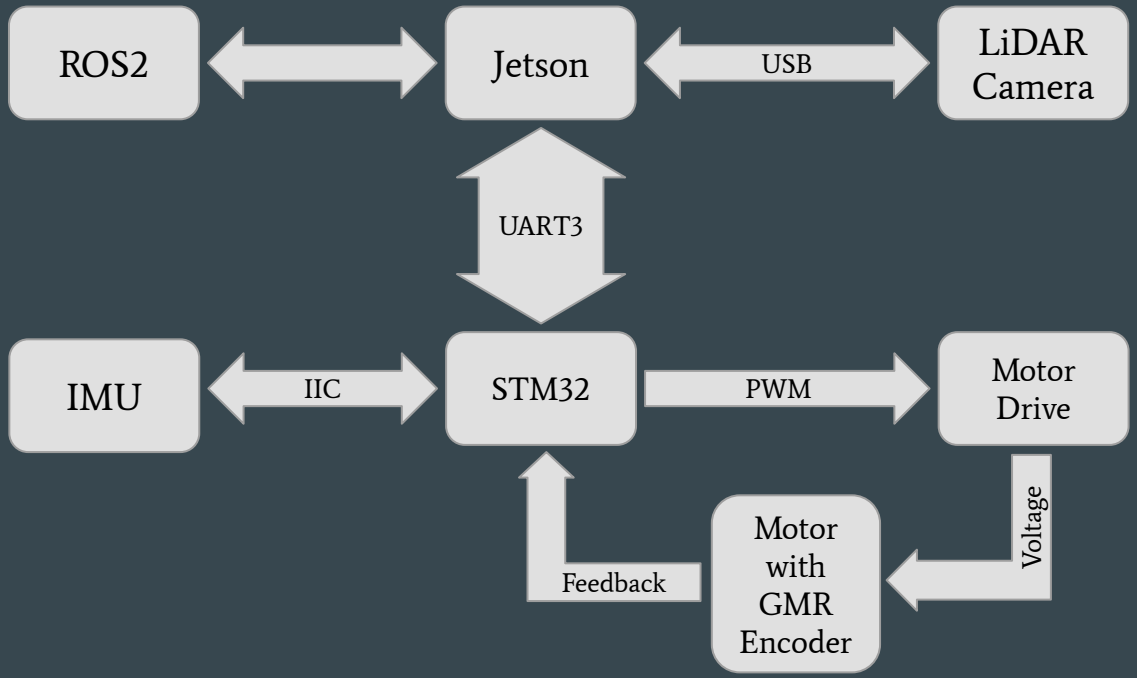
By: Kent

COMPUTE | Lesson Learnt

- Have difficulty to implement the algorithm from NAV2 into the real robot because there is some configuration that difference between using simulation and implement directly to the robot.
- Facing a lot of error that make the robot unable to navigate.
- Some configuration parameter need to be changed from previously using parameter in simulation in order for the robot to navigate in its environment.

By: Kent

MACHINE | STRUCTURE



Main Control Unit
STM32

Driver Unit
Motor Drive
Motor with GMR Encoder
Mecanum Wheel

Computing Unit
Jetson Orin/Jetson Nano

Sensors
Stereo Camera/RGB Camera
LiDAR
IMU
Accelerometers
Gyroscopes

By: Chen Junjie

MACHINE | REASONS FOR CHOICE

Main Control Unit
STM32: Extensible, rich interface, high performance, low power consumption

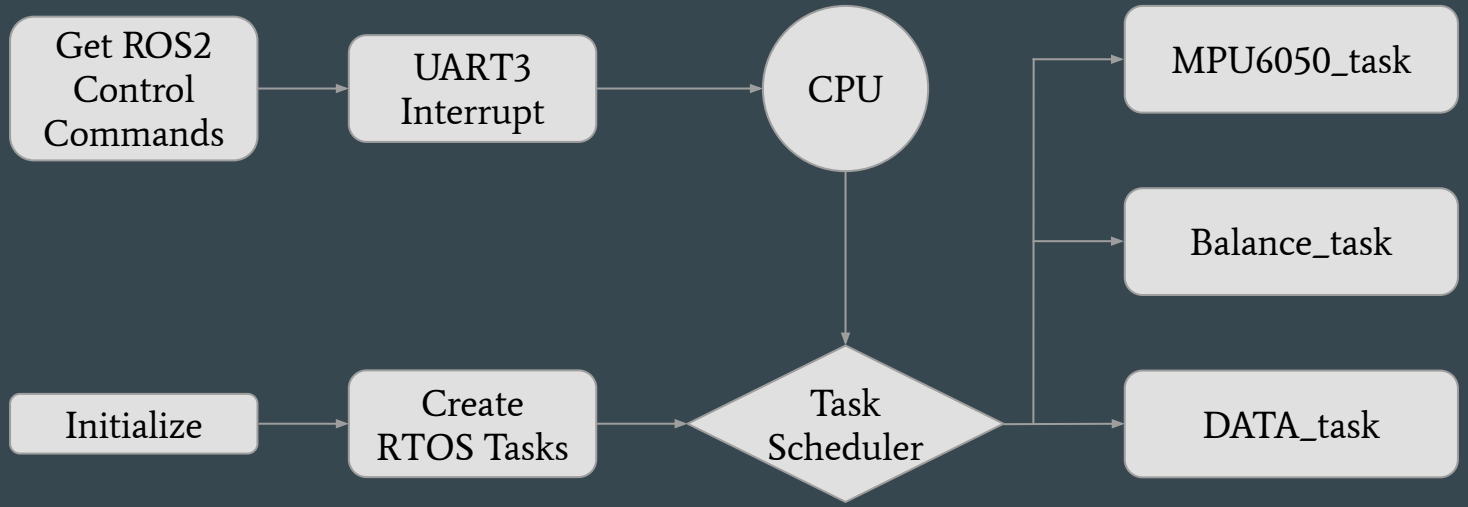
Driver Unit
Motor Drive: Mecanum wheels need 4-way drive
Motor with GMR Encoder: Enough speed and load capacity, high accuracy (500 ppr)
Mecanum Wheel: Omni-directional movement, flexibility in tight spaces

Computing Unit
Jetson Orin/Jetson Nano: Powerful computational performance, stronger support for AI, better ROS2 compatibility
ROS2: More reliably, saving resources, better real-time performance

Sensors
Stereo Camera/RGB Camera: Richer texture information, performs better in larger scale and dynamic environments
LiDAR: More accurate, smaller computational requirements, performs better in static and simple environments
IMU/Accelerometers/Gyroscopes: For measuring internal robot parameters used in localization

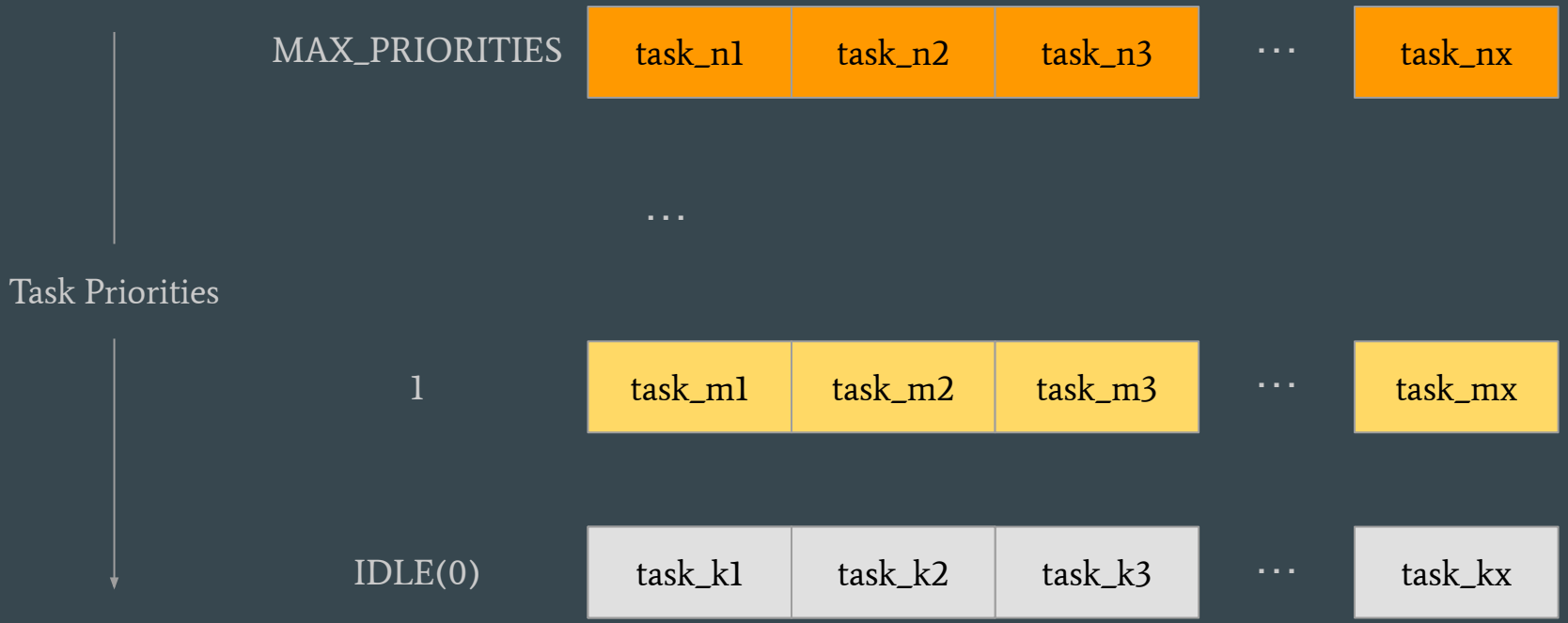
By: Chen Junjie

MACHINE | ACTUATOR



By: Chen Junjie

MACHINE | ACTUATOR



By: Chen Junjie

MACHINE | TASKS

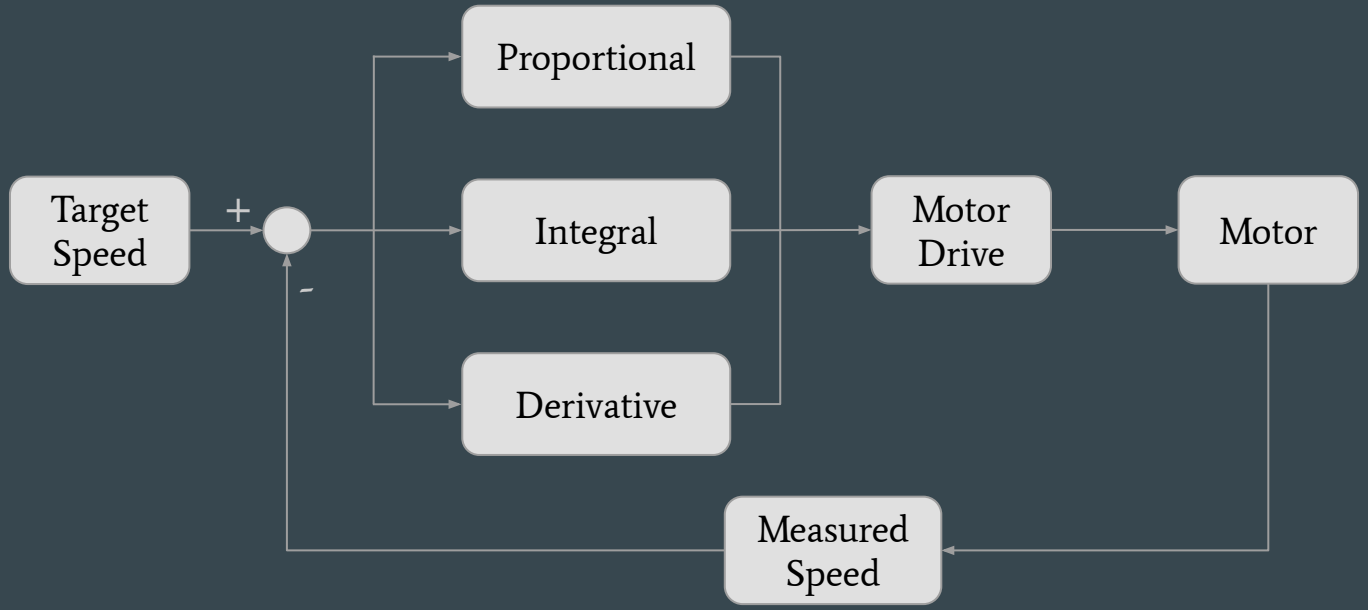
```
void MPU6050_task(void *pvParameters)
{
  // Get the current timestamp of the system tick counter
  u32 lastWakeTime = getSysTickCnt();

  while (1)
  {
    // Run the task at a frequency of 100Hz
    vTaskDelayUntil(&lastWakeTime,
F2T(RATE_100_HZ));

    // Read the gyroscope zero before starting
    if (Deviation_Count < CONTROL_DELAY)
    {
      Deviation_Count++;
      memcpy(Deviation_gyro, gyro, sizeof(gyro));
      memcpy(Deviation_accel, accel, sizeof(accel));
    }

    // Get gyroscope data
    MPU_Get_Gyroscope();

    // Get accelerometer data (raw values)
    MPU_Get_Accelscope();
  }
}
```

```
void Balance_task(void *pvParameters) {
  // Get the current timestamp of the system tick counter
  u32 lastWakeTime = getSysTickCnt();

  while (1) {
    // Run the task at a frequency of 100Hz
    vTaskDelayUntil(&lastWakeTime, F2T(RATE_100_HZ));

    // Get encoder data (real-time wheel speed)
    Get_Velocity_Form_Encoder();

    // Integrate velocity to get position
    POSITION_A += MOTOR_A.Encoder;
    .....

    // If the enable switch is in the ON position
    if (Turn_Off(Voltage) == 0) {
      // Position PID controller
      POSITION_A.Motor_Pwm = Position_PID(POSITION_A, POSITION_A.Target);
      Limit_A = Limit(POSITION_A.Motor_Pwm, abs(MOTOR_A.Target));
      .....

      // Speed closed-loop control to calculate the PWM value of each motor
      MOTOR_A.Motor_Pwm = Incremental_PID_A(MOTOR_A.Encoder, Limit_A);
      MOTOR_A.Motor_Pwm = Limit(MOTOR_A.Motor_Pwm, Threshold);
      .....

      // Set PWM control polarity
      Set_Pwm(MOTOR_A.Motor_Pwm, -MOTOR_B.Motor_Pwm,
-MOTOR_C.Motor_Pwm, MOTOR_D.Motor_Pwm, 0);
    }
    // Otherwise, do not allow the car to move, set PWM values to 0
    else  Set_Pwm(0, 0, 0, 0, 0);
  }
}
```

```
void data_task(void *pvParameters)
{
  // Get the current timestamp of the system tick counter
  u32 lastWakeTime = getSysTickCnt();

  while (1)
  {
    // Run the task at a frequency of 20Hz
    vTaskDelayUntil(&lastWakeTime,
F2T(RATE_20_HZ));

    // Assign the data to be sent
    data_transition();

    // Send data via USART3
    USART3_SEND();
  }
}
```

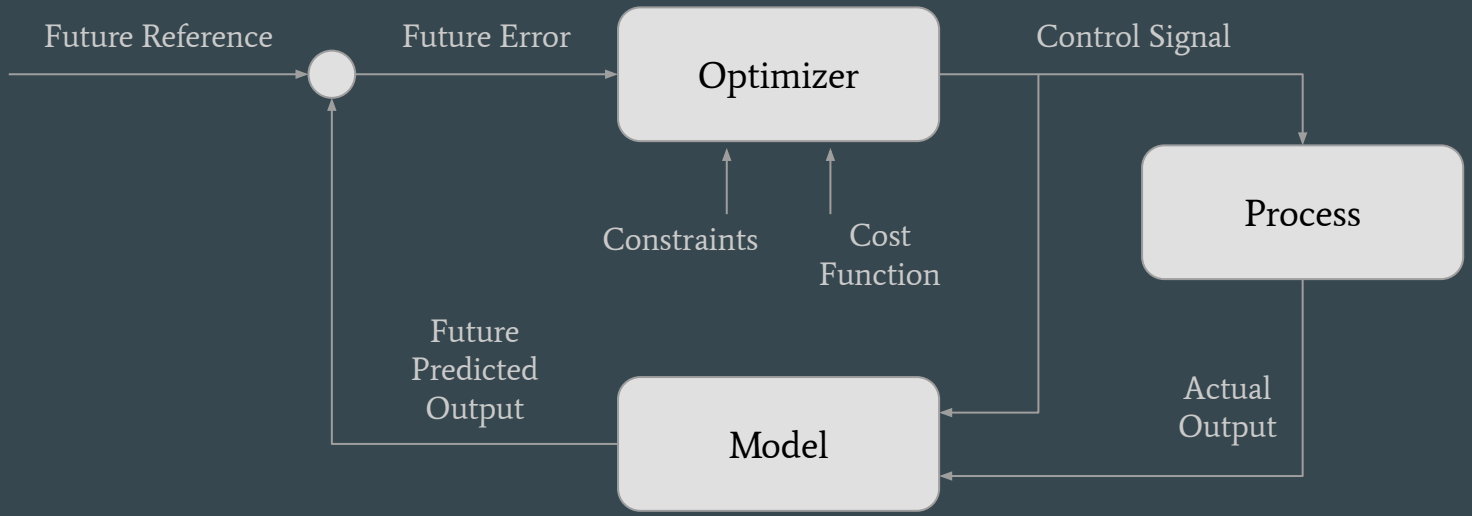
By: Chen Junjie

MACHINE | CONTROLLER SELECTION - SINGLE LOOP PID



By: Chen Junjie

MACHINE | CONTROLLER SELECTION - MPC



By: Chen Junjie

MACHINE | CONTROLLER SELECTION - MPC

Forward:

$$V_x = \frac{V_A + V_B + V_C + V_D}{4}$$

$$V_y = \frac{-V_A + V_B + V_C - V_D}{4}$$

$$V_w = \frac{-V_A + V_B - V_C + V_D}{2H + 2W}$$

Inverse:

$$V_A = V_x - V_y - V_w \cdot \left(\frac{W}{2} + \frac{H}{2}\right)$$

$$V_B = V_x + V_y + V_w \cdot \left(\frac{W}{2} + \frac{H}{2}\right)$$

$$V_C = V_x + V_y - V_w \cdot \left(\frac{W}{2} + \frac{H}{2}\right)$$

$$V_D = V_x - V_y + V_w \cdot \left(\frac{W}{2} + \frac{H}{2}\right)$$

State:

$$\begin{bmatrix} p_x \\ p_y \\ \phi \\ v_x \\ v_y \\ w \end{bmatrix}$$

Input:

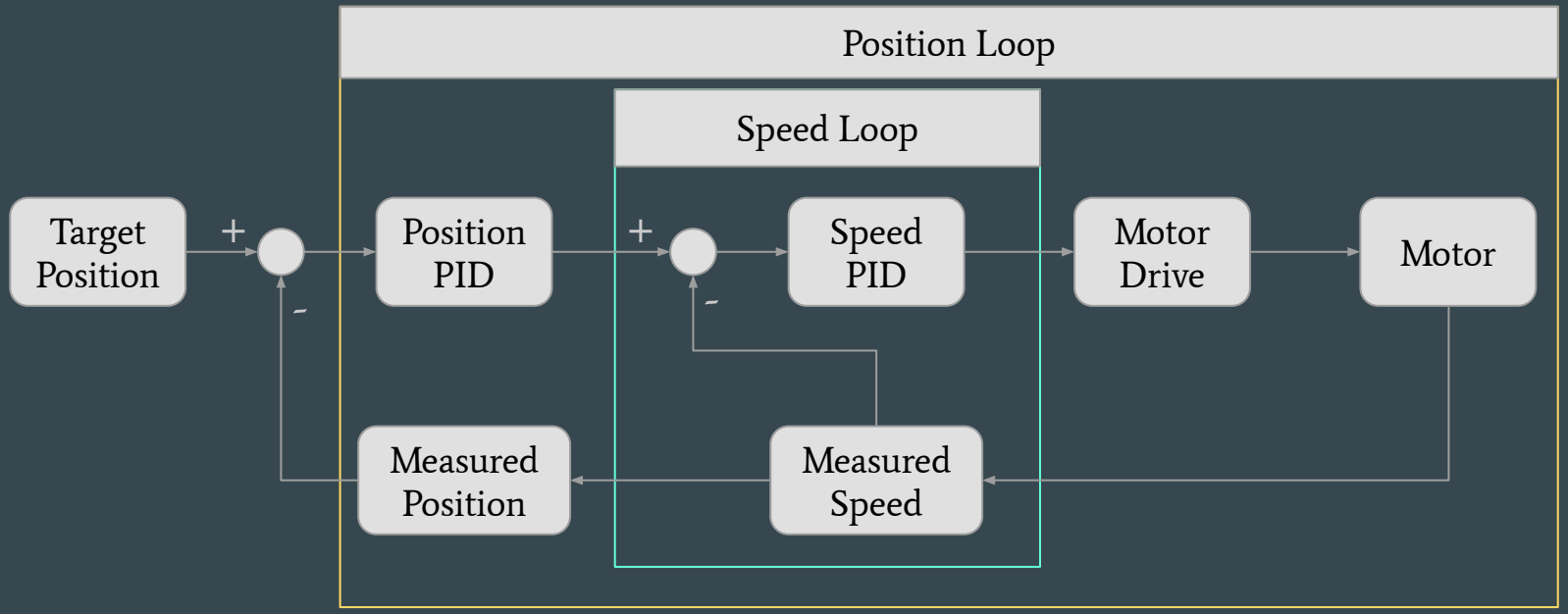
$$\begin{bmatrix} V_A \\ V_B \\ V_C \\ V_D \end{bmatrix}$$

State Space Function:

$$\begin{cases} \dot{p}_x = \dfrac{V_A + V_B + V_C + V_D}{4} \\ \dot{p}_y = \dfrac{-V_A + V_B + V_C - V_D}{4} \\ \dot{\phi} = \dfrac{-V_A + V_B - V_C + V_D}{2H + 2W} \\ \dot{v}_x = ? \\ \dot{v}_y = ? \\ \dot{w} = ? \end{cases}$$

By: Chen Junjie

MACHINE | CONTROLLER SELECTION - DUAL LOOP PID



By: Chen Junjie

MACHINE | Balance_task

```
// Update velocity
Encoder = Get_Velocity_Form_Encoder();

// Integrate velocity to obtain position
Position += Encoder;

// Position PID controller
Position_Motor = Position_PID(Position,
Target_Position);

// Limit the output of the position loop
Limit_A = Limit(Position_Motor,
abs(Target_Velocity));

// Velocity PID controller
Motor = Incremental_PID(Encoder, Limit_A);

// Limit the PWM output
Motor = Limit(Motor, Threshold);

// Set PWM output
Set_Pwm(Motor);
```

Pwm = Kp*e(k) + Ki*∑e(k) + Kd*[e(k)-e(k-1)]

```
int Position_PID(int Encoder, int Target) {
    static float Bias, Pwm, Integral_bias, Last_Bias;

    // Calculate the deviation
    Bias = Target - Encoder;

    // Integrate the deviation
    Integral_bias += Bias;

    // PID controller output calculation
    Pwm = Position_KP * Bias + Position_KI *
Integral_bias + Position_KD * (Bias - Last_Bias);

    // Save the last deviation for the next iteration
    Last_Bias = Bias;

    // Output the calculated PWM value
    return Pwm;
}
```

Pwm += Kp*[e(k)-e(k-1)] + Ki*e(k) + Kd*[e(k)-2*e(k-1)+e(k-2)]

```
int Incremental_PID(int Encoder, int Target) {
    static float Bias, Pwm, Last_bias;

    // Calculate the deviation
    Bias = Encoder - Target;

    // Incremental PID controller calculation
    Pwm += Velocity_KP * (Bias - Last_bias) +
Velocity_KI * Bias + Velocity_KD * (Bias - 2 *
Last_bias + Last_last_bias);

    // Save the last two deviations for the next iteration
    Last_last_bias = Last_bias
    Last_bias = Bias;

    // Incremental output
    return Pwm;
}
```

By: Chen Junjie

MACHINE | TASKS

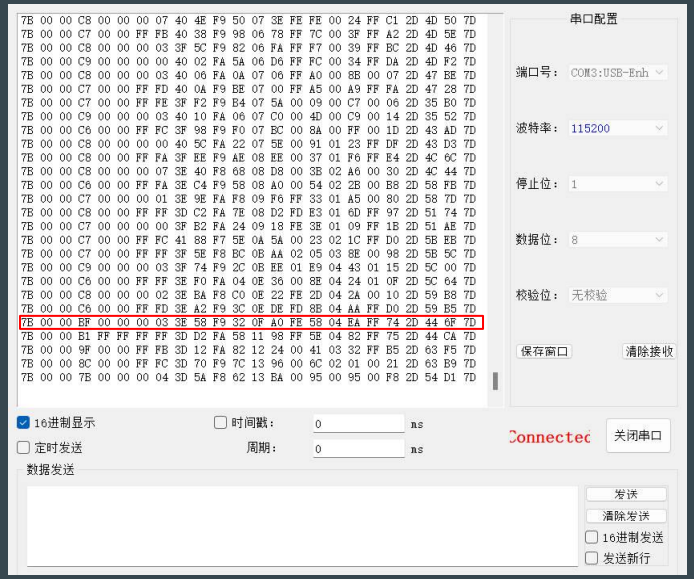
```
void MPU6050_task(void *pvParameters)
{
  // Get the current timestamp of the system tick counter
  u32 lastWakeTime = getSysTickCnt();

  while (1)
  {
    // Run the task at a frequency of 100Hz
    vTaskDelayUntil(&lastWakeTime,
F2T(RATE_100_HZ));

    // Read the gyroscope zero before starting
    if (Deviation_Count < CONTROL_DELAY)
    {
      Deviation_Count++;
      memcpy(Deviation_gyro, gyro, sizeof(gyro));
      memcpy(Deviation_accel, accel, sizeof(accel));
    }

    // Get gyroscope data
    MPU_Get_Gyroscope();

    // Get accelerometer data (raw values)
    MPU_Get_Accelscope();
  }
}
```

```
void Balance_task(void *pvParameters) {
  // Get the current timestamp of the system tick counter
  u32 lastWakeTime = getSysTickCnt();

  while (1) {
    // Run the task at a frequency of 100Hz
    vTaskDelayUntil(&lastWakeTime, F2T(RATE_100_HZ));

    // Get encoder data (real-time wheel speed)
    Get_Velocity_Form_Encoder();

    // Integrate velocity to get position
    POSITION_A += MOTOR_A.Encoder;
    .....

    // If the enable switch is in the ON position
    if (Turn_Off(Voltage) == 0) {
      // Position PID controller
      POSITION_A.Motor_Pwm = Position_PID(POSITION_A, POSITION_A.Target);
      Limit_A = Limit(POSITION_A.Motor_Pwm, abs(MOTOR_A.Target));
      .....

      // Speed closed-loop control to calculate the PWM value of each motor
      MOTOR_A.Motor_Pwm = Incremental_PID_A(MOTOR_A.Encoder, Limit_A);
      MOTOR_A.Motor_Pwm = Limit(MOTOR_A.Motor_Pwm, Threshold);
      .....

      // Set PWM control polarity
      Set_Pwm(MOTOR_A.Motor_Pwm, -MOTOR_B.Motor_Pwm,
-MOTOR_C.Motor_Pwm, MOTOR_D.Motor_Pwm, 0);
    }
    // Otherwise, do not allow the car to move, set PWM values to 0
    else  Set_Pwm(0, 0, 0, 0, 0);
  }
}
```

```
void data_task(void *pvParameters)
{
  // Get the current timestamp of the system tick counter
  u32 lastWakeTime = getSysTickCnt();

  while (1)
  {
    // Run the task at a frequency of 20Hz
    vTaskDelayUntil(&lastWakeTime,
F2T(RATE_20_HZ));

    // Assign the data to be sent
    data_transition();

    // Send data via USART3
    USART3_SEND();
  }
}
```

By: Chen Junjie

MACHINE | DATA_task



7B	00	00	BF	00	00	00	03
head (0X7B)	flag_ stop	Vx		Vy		Vz	
3E	58	F9	32	0F	A0	FE	58
Ax		Ay		Az		wx	
04	EA	FF	74	2D	44	6F	7D
wy		wz		Voltage		check digit	tail (0X7D)

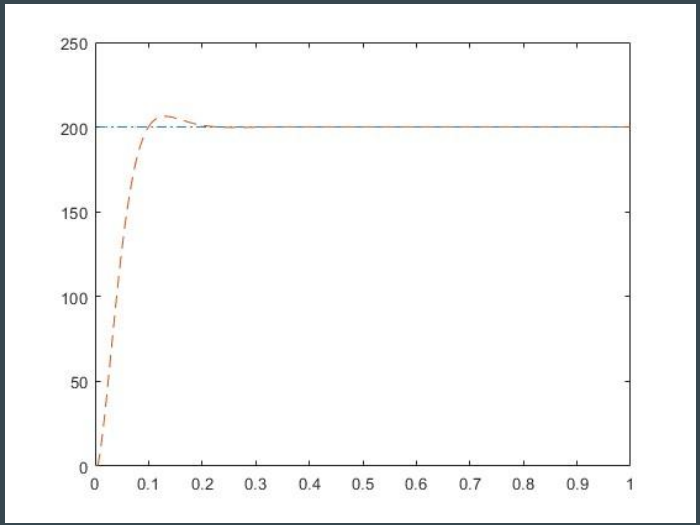
Vx: 00 BF = 0000 0000 1011 1111 ➔ Vx = (0*16+0)*256+(B*16+F) = (0*16+0)*256+(11*16+15) = 191 mm/s

$$V_A = V_x - V_y - V_w \cdot \left(\frac{W}{2} + \frac{H}{2}\right)$$

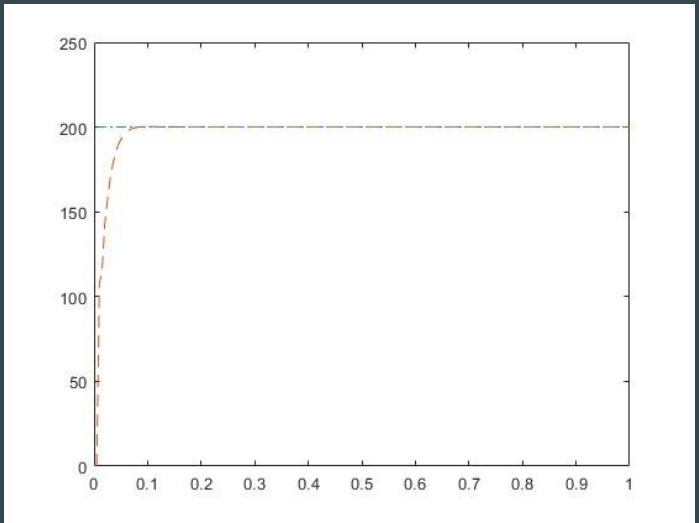
By: Chen Junjie

MACHINE | Balance_task

Single Loop PID



Dual Loop PID



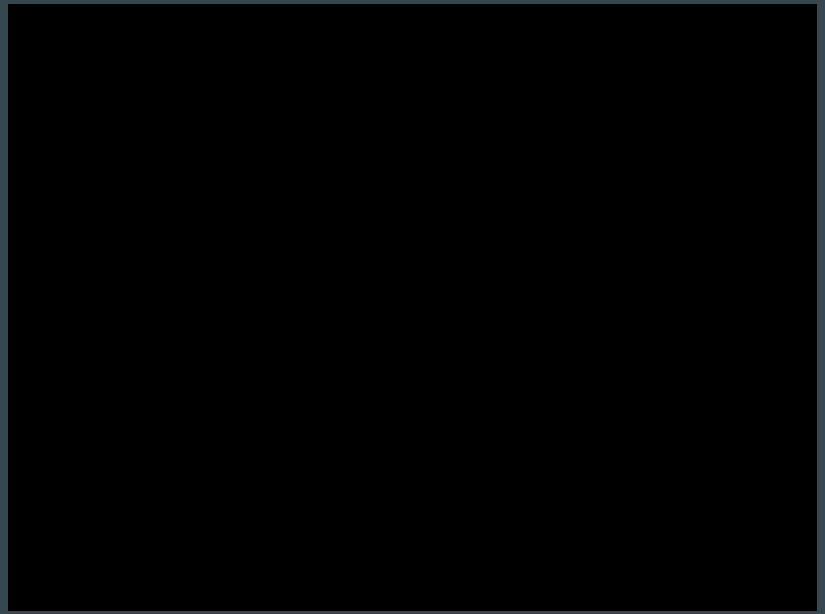
By: Chen Junjie

MACHINE | Lesson Learnt

- For PID control, positional PID requires integral limits and output limits, while incremental PID requires only output limits.
- Be sure to disconnect the power first when connecting accessories to avoid touching the contacts on the bottom of the STM32
- STM32 controllers are more convenient to download programs through the serial port than the SWD interface
- Pay attention to details and don't make mistakes when building the model, especially at the very beginning

By: Chen Junjie

Demo Video



Disclaimer: Recorded with our smaller robot for portability

Futurework

- ME 5400B : Extend base with tablet (for display and sensors) mounted on flat-collapsible telescoping arm for manipulation (to press elevator buttons)
- Revisit pure multi camera approach - Matic Robot Vacuum
- GRIP Startup

By: Benjamin

Thank You