Node: Position_approximator_mecanum

Header

Team: Groep 4 RMB

Created: 02.12.2025 by Rik van Velzen

name	date	Short description
Position_approximator_mecanum	18.11.2025	This node subscribes to mecanum wheel velocity messages, computes the mecanum formulas, integrates velocities over time, and publishes a PositionData message containing the estimated robot pose (x, y, yaw).

Node description

This node estimates the robot’s position and orientation using mecanum wheel odometry.
It listens to wheel angular velocities, converts them into linear and angular robot velocities, integrates these values over time, and publishes the resulting pose.

The node allows configuration of wheel radius, robot dimensions, start pose, and topic names through ROS2 parameters.

Node sub-objects and functions (communication objects):

(timer, publisher, subscriber, service server, action server, service client, action client

publisher : mecanum_pub_ Publishes a PositionData message containing the computed x, y, z, and yaw_z values on a configurable topic.
Publisher function: publish(PositionData) Sends the new pose estimate after each velocity update cycle.

subscriber : mecanum_sub_ Subscribes to a mecanum msg topic. And receives angular wheel velocities (wfl, wfr, wrl, wrr) from another node to compute odometry.
Callback function: calculate_mecanum Processes wheel velocities, computes robot velocity (vx, vy, yaw_vz), integrates pose, and publishes updated position.

Node actions, messages and services:

Messages:

- Published Message Type: g425_assign4_interfaces_pkg::msg::PositionData
- Subscribed Message Type: g425_assign4_interfaces_pkg::msg::Mecanum

Topics:

- mecanum_position: Calculated position data for storage
- mecanum_velocity: simulated mecanum velocity data

Custom Node functions :

Custom function: declare_parameters()

Declares and retrieves ROS2 parameters for wheel radius, robot size, start pose, and topic names. Stores them in class variables for later use.

Custom function: calculate_mecanum()

Main odometry computation function.
Converts timestamps to compute time delta, calculates robot velocities from wheel speeds using mecanum kinematics, integrates these to update robot pose, logs data, and publishes the updated PositionData message.

Node implementation (main):

This node follows a standard single-threaded ROS2 node implementation.

No timers, services, multithreading, or action servers are used.

The only added logic is the exclusion of “main()” and “private:” during unit tests using the macro TESTING_EXCLUDE_MAIN.

Standard implementation:

```
int main(int argc, char **argv)
{
  rclcpp::init(argc, argv);
  auto node = std::make_shared<position_approximator_mecanum>();

  rclcpp::spin(node);

  rclcpp::shutdown();

  return 0;

}
```

Node dependencies :

g425_assign4_interfaces_pkg/msg/mecanum.hpp: Custom message used to receive mecanum wheel velocity data that is simulated on another node.

g425_assign4_interfaces_pkg/msg/position_data.hpp: Custom message used to send position data so it can be put into a database and be visualized.

rclcpp: Standard ROS2 C++ client library for node, subscription, and publisher functionality.