

Báo cáo bài giữa kỳ môn Lập trình Robot với ROS

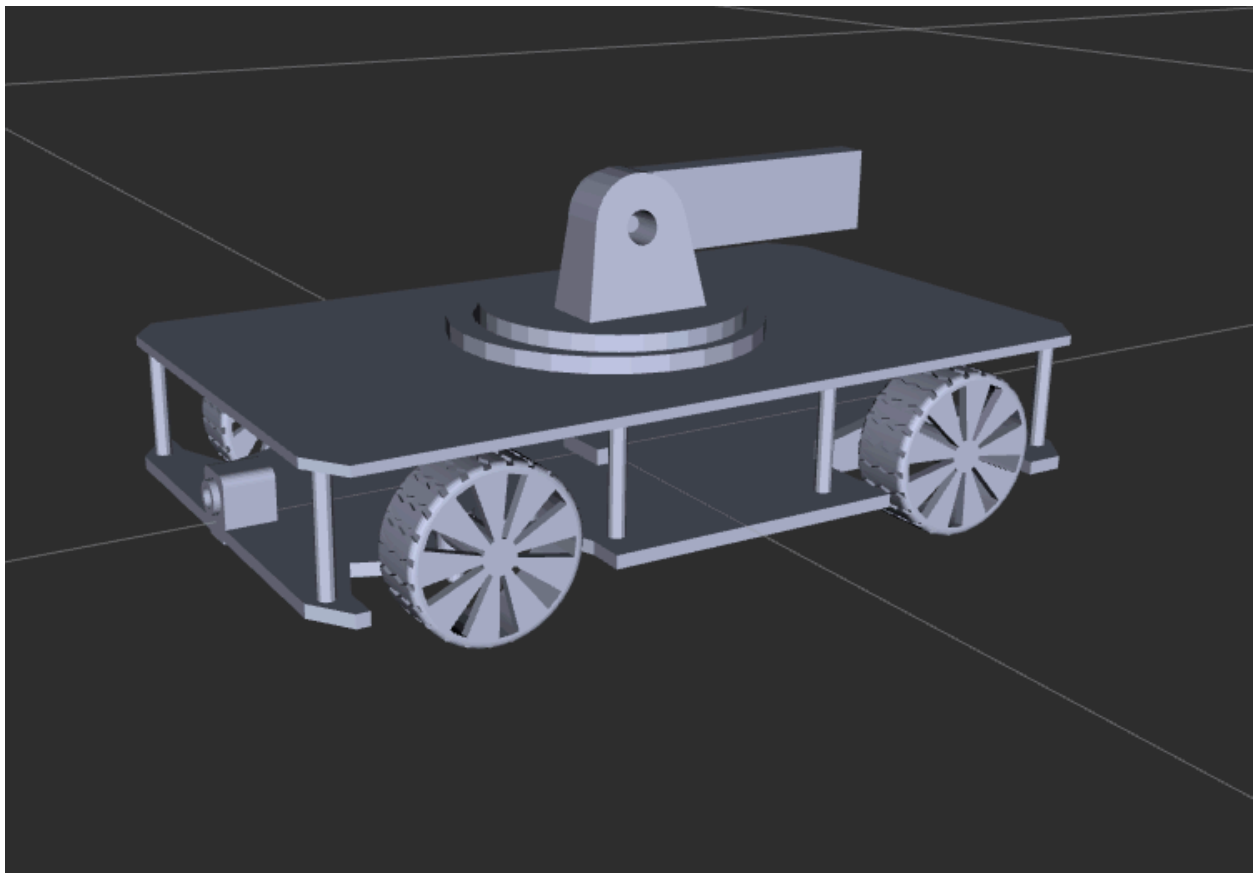
Giảng viên: TS. Lê Xuân Lực
KS. Dương Văn Tân

Họ và tên: Nguyễn Anh Vũ
Mã sinh viên: 22027552

1. Giới thiệu về chủ đề

Mô phỏng robot trong môi trường ảo đã trở thành một công cụ quan trọng trong việc phát triển và thử nghiệm các hệ thống robot mà không cần đến phần cứng thực tế, giúp giảm chi phí và thời gian nghiên cứu. Dự án này tập trung vào việc thiết kế và mô phỏng một robot di động tích hợp tay máy sử dụng ROS (Robot Operating System) và Gazebo, hai nền tảng phổ biến trong lĩnh vực robot học. Robot được mô phỏng là một hệ thống kết hợp giữa khả năng di chuyển linh hoạt trên mặt phẳng và thao tác vật thể thông qua tay máy, cùng với các cảm biến hiện đại như IMU (Inertial Measurement Unit), camera và encoder. Mục tiêu của dự án là xây dựng một mô hình robot hoàn chỉnh trong môi trường Gazebo, cho phép kiểm tra động học, điều khiển và tích hợp cảm biến, từ đó cung cấp nền tảng cho các ứng dụng thực tế như điều hướng tự động, xử lý hình ảnh và thao tác trong không gian 3D.

2. Dạng robot



Hình 1: hình dạng tổng quát của robot

Robot trong dự án này là một robot di động 4 bánh kết hợp tay máy 2 bậc tự do, được định nghĩa chi tiết trong file URDF (Unified Robot Description Format). Cụ thể:

- **Phần di động:** Robot sử dụng cấu hình skid-steer với 4 bánh xe thông qua các khớp liên tục. Hai bánh trước được gắn vào các trục trung gian, trong khi hai bánh sau gắn trực tiếp vào thân. Cấu hình này cho phép robot di chuyển linh hoạt trên mặt phẳng bằng cách điều chỉnh tốc độ và hướng quay của từng bánh xe.
- **Tay máy:** Robot được trang bị một tay máy gồm hai phần: `arm_base_link` (đế tay máy) và `arm_link` (đoạn tay máy), kết nối với nhau qua khớp quay (revolute joint). Đế tay máy được gắn vào thân robot thông qua khớp quay (`arm_base_joint`) với trục quay theo hướng X, trong khi đoạn tay máy nối với đế qua khớp quay (`arm_joint`) với trục quay theo hướng Y. Cấu trúc này cung cấp 2 bậc tự do, cho phép tay máy thực hiện các chuyển động xoay trong không gian 3D.
- **Cảm biến:** Robot tích hợp ba loại cảm biến chính: IMU (gắn cố định vào thân robot), camera (hỗ trợ thu thập hình ảnh môi trường), và encoder (đo vị trí và vận tốc bánh xe thông qua dữ liệu odometry).

3. Động học

Động học của robot được phân tích dựa trên hai thành phần chính: phần di động và tay máy.

- **Động học phần di động:** Robot sử dụng 4 bánh xe với plugin `gazebo_ros_skid_steer_drive.so`, tuân theo mô hình skid-steer. Robot có thể xoay tại nhưng không di chuyển ngang do đặc điểm của skid-steer.
- **Động học tay máy:** Tay máy có 2 bậc tự do với hai khớp quay: `arm_base_joint` và `arm_joint`.

4. Kích thước

Kích thước của robot được suy ra từ các thông số trong URDF:

- **Thân robot (car):** Chiều dài khoảng 0.155 m, chiều rộng khoảng 0.102 m, chiều cao khoảng 0.048 m (từ `arm_base_joint`).
- **Bánh xe:** Đường kính 0.1 m (theo plugin `skid_steer_drive`), độ dày cần xác định từ file STL.
- **Tay máy:** Đế (`arm_base_link`) dài khoảng 0.025 m theo trục X, đoạn tay (`arm_link`) cần đo từ STL. Chiều cao tối đa phụ thuộc vào góc quay của khớp.
- **Tổng thể:** Kích thước ước tính là 0.155 m (dài) x 0.102 m (rộng) x 0.1 m (cao) (chưa tính tay máy mở rộng).

5. Mô tả file URDF

File URDF là một tệp XML định nghĩa cấu trúc vật lý của robot, bao gồm 12 link và 11 joint. File được tạo từ SolidWorks bằng công cụ SolidWorks to URDF Exporter, chứa thông tin về hình học (file STL), quán tính, và các thẻ Gazebo để mô phỏng. Các thành phần chính bao gồm thân robot, bánh xe, tay máy, và cảm biến, được liên kết chặt chẽ để tạo thành một hệ thống hoàn chỉnh.

6. Liên kết của các link

Robot bao gồm 12 link, liên kết qua 11 joint:

- **Thân chính (car):** Gắn 4 bánh xe, tay máy, camera, và IMU.
- **Bánh xe:**
 - r_wheel_link/l_wheel_link nối với car qua khớp revolute (trục Z).
 - wheel1_link/wheel2_link nối với r_wheel_link/l_wheel_link qua khớp continuous (trục Y).
 - wheel3_link/wheel4_link nối trực tiếp với car qua khớp continuous (trục Y).
- **Tay máy:** arm_base_link nối với car qua khớp revolute (trục X), arm_link nối với arm_base_link qua khớp revolute (trục Y).
- **Cảm biến:** camera và IMU nối với car qua khớp fixed.

7. Các cảm biến

File URDF tích hợp ba cảm biến:

- **IMU:** Gắn vào link IMU, dùng plugin libgazebo_ros_imu_sensor.so, xuất dữ liệu qua topic /imu/data (tần số 100 Hz).
- **Camera:** Gắn vào link camera, dùng plugin libgazebo_ros_camera.so, xuất hình ảnh qua topic /camera/image_raw (640x480, 30 Hz).
- **Encoder:** Tích hợp qua bánh xe và plugin libgazebo_ros_skid_steer_drive.so, xuất dữ liệu odometry qua topic /odom.

8. Mô tả Gazebo

Gazebo là môi trường mô phỏng 3D tích hợp với ROS qua các plugin:

- **Plugin điều khiển:**
 - libgazebo_ros_skid_steer_drive.so: Điều khiển skid-steer cho 4 bánh xe qua topic /cmd_vel.
 - libgazebo_ros_joint_pose_trajectory.so: Điều khiển tay máy qua topic /car2/arm_trajectory_command.
- **Plugin cảm biến:** libgazebo_ros_imu_sensor.so (IMU) và libgazebo_ros_camera.so (camera).
- **Vật lý:** Sử dụng thông số quán tính và va chạm từ URDF, kết hợp file STL để mô phỏng chuyển động và tương tác.
- **Ứng dụng:** Robot được chạy bằng lệnh roslaunch car2 gazebo.launch, cho phép điều khiển và thu thập dữ liệu thực tế.

9. Mô tả cơ chế điều khiển trên Gazebo

Cơ chế điều khiển của robot trong Gazebo được thực hiện thông qua các plugin tích hợp trong file URDF, kết nối với ROS để nhận lệnh và phản hồi dữ liệu. Các thành phần điều khiển chính bao gồm:

- **Điều khiển phần di động (Skid-Steer):**
 - Plugin: libgazebo_ros_skid_steer_drive.so.
 - Cơ chế: Plugin này điều khiển 4 bánh xe (wheel1_joint, wheel2_joint, wheel3_joint, wheel4_joint) dựa trên vận tốc tuyến tính và góc được gửi qua topic /cmd_vel (dạng geometry_msgs/Twist). Các thông số chính:
 - Khoảng cách giữa bánh xe: 0.3 m.
 - Đường kính bánh xe: 0.1 m.
 - Lực mô-men xoắn tối đa: 10.0 Nm.
 - Hoạt động: Khi nhận lệnh từ /cmd_vel, plugin tính toán vận tốc riêng lẻ cho từng bánh xe để đạt được chuyển động mong muốn (tiến, lùi, xoay). Dữ liệu odometry được xuất qua topic /odom, cung cấp thông tin vị trí và hướng của robot.
- **Điều khiển tay máy:**
 - Plugin: libgazebo_ros_joint_pose_trajectory.so.
 - Cơ chế: Plugin này điều khiển hai khớp của tay máy (arm_base_joint và arm_joint) bằng cách nhận lệnh vị trí từ topic /car2/arm_trajectory_command. Các thông số:
 - Tần số cập nhật: 100 Hz.
 - Giới hạn góc: arm_base_joint (-3.14 đến 3.14 rad), arm_joint (-3.14 đến 0 rad).
 - Hoạt động: Lệnh vị trí (pose) được gửi dưới dạng danh sách góc khớp, sau đó Gazebo mô phỏng chuyển động của tay máy theo quỹ đạo được chỉ định.
- **Tích hợp ROS:**
 - Các plugin sử dụng ROS để giao tiếp giữa Gazebo và các node điều khiển bên ngoài. Ví dụ, lệnh điều khiển có thể được gửi từ script Python (như move_robot.py hoặc move_arm.py) hoặc trực tiếp qua lệnh rostopic pub.
 - Tần số cập nhật cao (100 Hz cho skid-steer và tay máy, 30 Hz cho camera) đảm bảo phản hồi nhanh và chính xác trong mô phỏng.
- **Ứng dụng thực tế:** Cơ chế này cho phép điều khiển robot thông qua bàn phím (WASD cho di chuyển, IJKL cho tay máy) hoặc các thuật toán tự động, mô phỏng hành vi thực tế của robot trong môi trường ảo.

10. Các thành phần chính của code, structure folder dự án

Dự án được xây dựng trong ROS, với cấu trúc thư mục và các thành phần code được tổ chức để hỗ trợ mô phỏng và điều khiển robot. Dưới đây là mô tả chi tiết:

Các thành phần chính của code

- **File URDF (car2.urdf):**
 - Định nghĩa cấu trúc robot, bao gồm link, joint, và plugin Gazebo.
 - Đường dẫn: car2/urdf/car2.urdf.

- **File Launch:**
 - gazebo.launch: Khởi chạy Gazebo với mô hình robot từ URDF, tải môi trường mô phỏng và các plugin.
 - display.launch: Khởi chạy RViz để hiển thị trạng thái robot, bao gồm TF (transform) và dữ liệu cảm biến.
 - Đường dẫn: car2/launch/.
- **Script điều khiển:**
 - move_robot.py: Script Python điều khiển chuyển động skid-steer của robot bằng bàn phím (WASD), gửi lệnh tới topic /cmd_vel.
 - move_arm.py: Script Python điều khiển tay máy bằng bàn phím (IJKL), gửi lệnh tới topic /car2/arm_trajectory_command.
 - Thư viện sử dụng: rospy, geometry_msgs, trajectory_msgs.
 - Đường dẫn: car2/scripts/.
- **File cấu hình:**
 - CMakeLists.txt và package.xml: Định nghĩa package ROS, liệt kê các phụ thuộc (như gazebo_ros, ros_control) và quy tắc build.
 - Đường dẫn: car2/.

Structure folder dự án

Cấu trúc thư mục của package car2 được tổ chức theo chuẩn ROS:

- **car2/:**
 - urdf/:
 - car2.urdf: File mô tả robot.
 - meshes/:
 - Các file STL (ví dụ: car.STL, wheel1_link.STL, arm_link.STL, v.v.): Định nghĩa hình học của các link.
 - launch/:
 - gazebo.launch: Khởi chạy mô phỏng Gazebo.
 - display.launch: Khởi chạy RViz.
 - scripts/:
 - move_robot.py: Điều khiển phần di động.
 - move_arm.py: Điều khiển tay máy.
 - CMakeLists.txt: Quy tắc build package.
 - package.xml: Thông tin package và phụ thuộc.

Các package phụ thuộc

- gazebo_ros, gazebo_plugins, gazebo_ros_control, ros_control, ros_controllers, robot_state_publisher, joint_state_publisher, tf.
- Được khai báo trong package.xml để đảm bảo tích hợp đầy đủ.

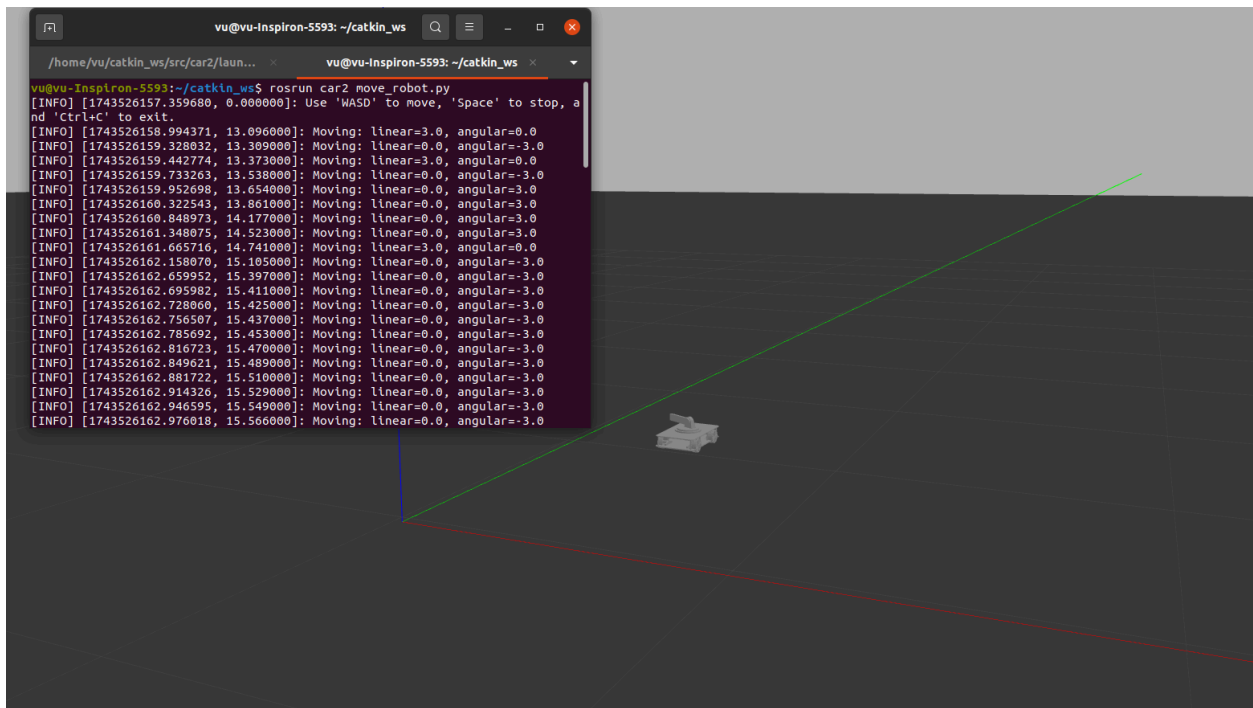
Quy trình hoạt động

1. Build: Chạy `catkin_make` trong thư mục `~/catkin_ws` để biên dịch package.
2. Source: Thêm source `~/catkin_ws/devel/setup.bash` vào `.bashrc` để ROS nhận diện package.
3. Chạy: Sử dụng các file launch và script để khởi động mô phỏng và điều khiển robot.

Các chức năng

1. Di chuyển sử dụng phím WASD

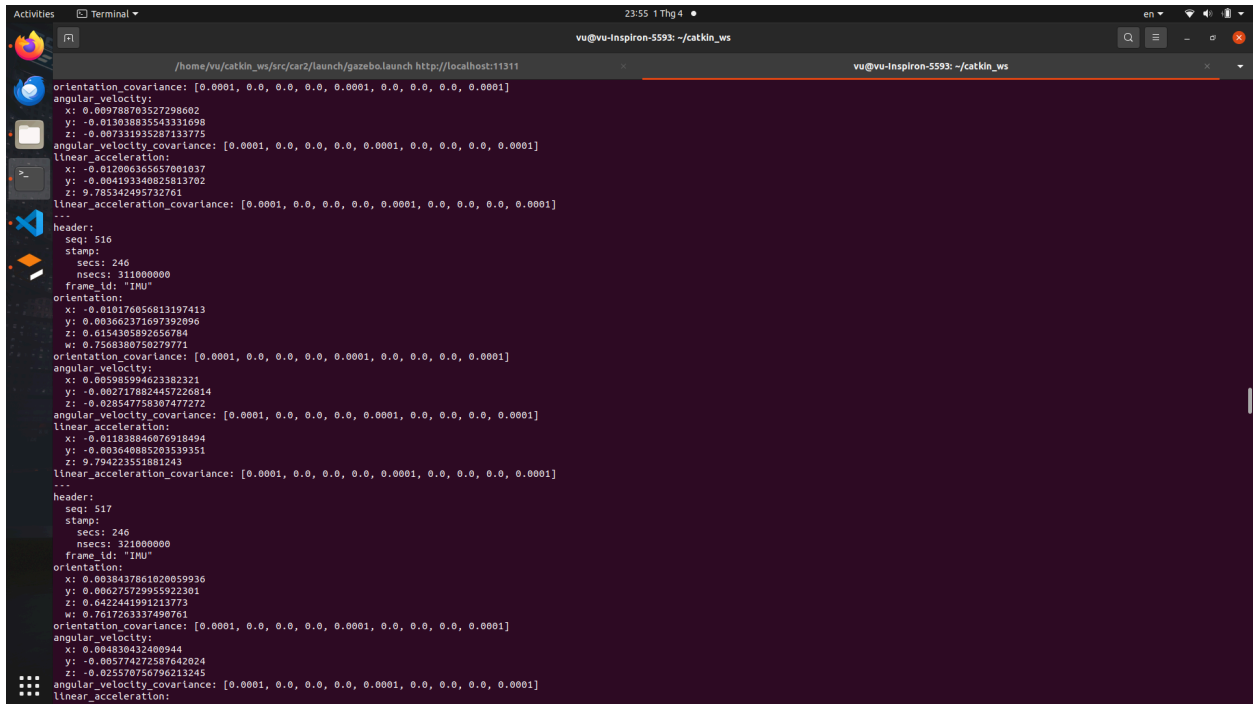
`roslaunch car2 move_robot.py`



2. Chuyển động tay máy sử dụng IJKL

`roslaunch car2 move_robot.py`

rostopic echo /imu/data

A terminal window titled 'vu@vu-Inspiron-5593: ~/catkin_ws' displays the output of the command 'rostopic echo /imu/data'. The output shows a series of IMU data messages. The first message includes orientation_covariance, angular_velocity, angular_velocity_covariance, linear_acceleration, and linear_acceleration_covariance. The second message includes a header with sequence number 516, timestamp 246, and frame ID 'IMU', followed by orientation, orientation_covariance, angular_velocity, angular_velocity_covariance, linear_acceleration, and linear_acceleration_covariance. The third message includes a header with sequence number 517, timestamp 246, and frame ID 'IMU', followed by orientation, orientation_covariance, angular_velocity, angular_velocity_covariance, linear_acceleration, and linear_acceleration_covariance. The terminal window has a dark background and a light-colored text. The output is as follows:

```
/home/vu/catkin_ws/src/car2/launch/gazebo.launch http://localhost:11311
vu@vu-Inspiron-5593: ~/catkin_ws
orientation_covariance: [0.0001, 0.0, 0.0, 0.0, 0.0001, 0.0, 0.0, 0.0, 0.0001]
angular_velocity:
  x: 0.009788703527298602
  y: -0.01303883554331698
  z: -0.007331925287113775
angular_velocity_covariance: [0.0001, 0.0, 0.0, 0.0, 0.0001, 0.0, 0.0, 0.0, 0.0001]
linear_acceleration:
  x: -0.0120603657001037
  y: -0.004193140025813702
  z: 9.785342495732761
linear_acceleration_covariance: [0.0001, 0.0, 0.0, 0.0, 0.0001, 0.0, 0.0, 0.0, 0.0001]
---
header:
  seq: 516
  stamp:
    secs: 246
    nsecs: 311000000
  frame_id: "IMU"
orientation:
  x: -0.010176056813197413
  y: 0.003662371697392096
  z: 0.0154305892656784
  w: 0.7568389758279771
orientation_covariance: [0.0001, 0.0, 0.0, 0.0, 0.0001, 0.0, 0.0, 0.0, 0.0001]
angular_velocity:
  x: 0.005985994623382321
  y: -0.0027178824457226814
  z: -0.028547758307477272
angular_velocity_covariance: [0.0001, 0.0, 0.0, 0.0, 0.0001, 0.0, 0.0, 0.0, 0.0001]
linear_acceleration:
  x: -0.01838846076918494
  y: -0.003640885203539351
  z: 9.79422351881243
linear_acceleration_covariance: [0.0001, 0.0, 0.0, 0.0, 0.0001, 0.0, 0.0, 0.0, 0.0001]
---
header:
  seq: 517
  stamp:
    secs: 246
    nsecs: 321000000
  frame_id: "IMU"
orientation:
  x: 0.0038437861020059936
  y: 0.006275729955922301
  z: 0.6422441991213773
  w: 0.761126337490761
orientation_covariance: [0.0001, 0.0, 0.0, 0.0, 0.0001, 0.0, 0.0, 0.0, 0.0001]
angular_velocity:
  x: 0.004838432400944
  y: -0.005774272587642024
  z: -0.025570756796213245
angular_velocity_covariance: [0.0001, 0.0, 0.0, 0.0, 0.0001, 0.0, 0.0, 0.0, 0.0001]
linear_acceleration:
```

5. Hoạt động của Encoder

rostopic echo /odom

```
Activities Terminal 23:56 1 Thg 4 en vu@vu-Inspiron-5593: ~/catkin_ws
/home/vu/catkin_ws/src/car2/launch/gazebo.launch http://localhost:11311 vu@vu-Inspiron-5593: ~/catkin_ws

x: -9.520352736041108e-05
y: 5.8416726852188834e-05
z: 0.645462868251539
w: 0.7637916493126098
covariance: [0.0001, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0001, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1000000000000.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1000000000000.0, 0
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.01]
twist:
  linear:
    x: 0.0019242814170107073
    y: -0.001477821171646827
    z: 0.0
  angular:
    x: 0.0
    y: 0.0
    z: -0.013342842582163027
covariance: [0.0001, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0001, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1000000000000.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1000000000000.0, 0
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.01]
...
header:
  seq: 28748
  stamp:
    secs: 287
    nsecs: 656000000
  frame_id: "odom"
  child_frame_id: "base_link"
pose:
  position:
    x: 0.41694130330261864
    y: 1.263913404927425
    z: 0.0
  orientation:
    x: -9.520352736041108e-05
    y: 5.8416726852188834e-05
    z: 0.645462868251539
    w: 0.7637916493126098
covariance: [0.0001, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0001, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1000000000000.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1000000000000.0, 0
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.01]
twist:
  linear:
    x: 0.0019242814170107073
    y: -0.001477821171646827
    z: 0.0
  angular:
    x: 0.0
    y: 0.0
    z: -0.013342842582163027
covariance: [0.0001, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0001, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1000000000000.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1000000000000.0, 0
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.01]
...

```