Working with the *robot_localization* Package

Tom Moore
Clearpath Robotics

What is *robot_localization*?

- **General purpose state estimation package**
- **No limit on the number of input data sources**
- **Supported message types for state estimation nodes**
  - **nav_msgs/Odometry**
  - **geometry_msgs/PoseWithCovarianceStamped**
  - **geometry_msgs/TwistWithCovarianceStamped**
  - **sensor_msgs/Imu**
- **State vector:** $\begin{bmatrix} x & y & z & \alpha & \beta & \gamma & \dot{x} & \dot{y} & \dot{z} & \dot{\alpha} & \dot{\beta} & \dot{\gamma} & \ddot{x} & \ddot{y} & \ddot{z} \end{bmatrix}$
- **Two typical use cases**
  - **Fuse continuous sensor data (e.g., wheel encoder odometry and IMU) to produce locally accurate state estimate**
  - **Fuse continuous data with global pose estimates (e.g., from SLAM) to provide an accurate and complete global state estimate**

Nodes

CLEARPATH ROBOTICS™

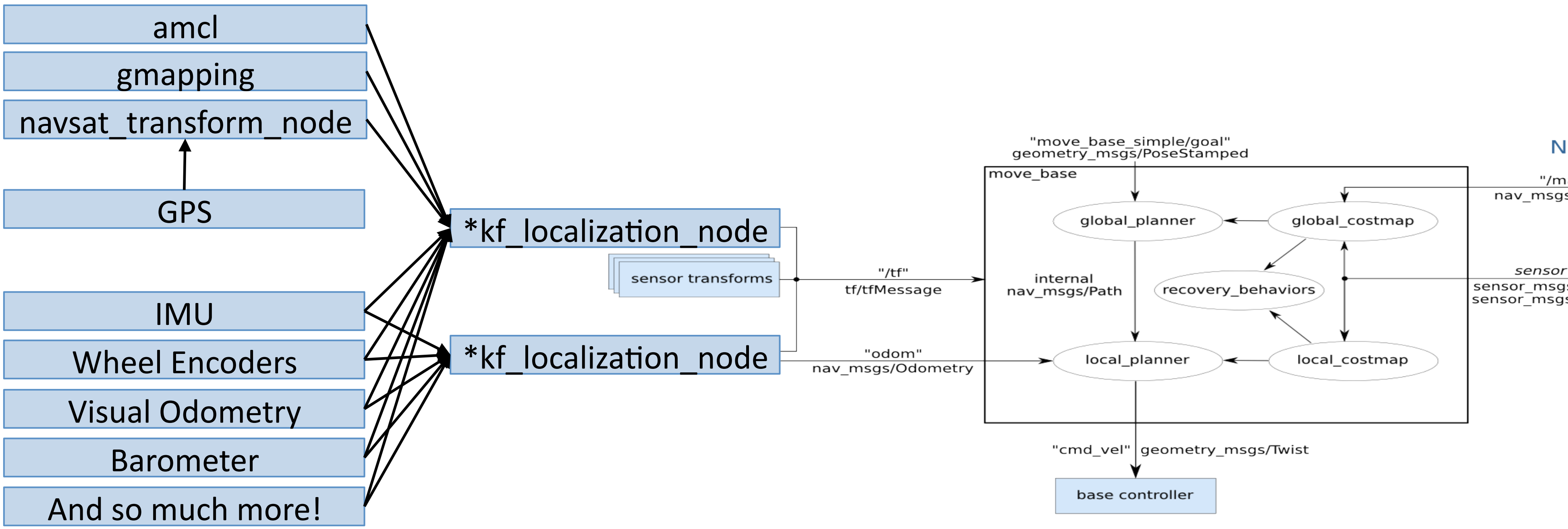
State estimation nodes

- *ekf_localization_node* – Implementation of an extended Kalman filter (EKF)
- *ukf_localization_node* – Implementation of an unscented Kalman filter (UKF)

Sensor preprocessing nodes

- *navsat_transform_node* – Allows users to easily transform geographic coordinates (latitude and longitude) into the robot's world frame (typically *map* or *odom*)

robot_localization and the ROS Navigation Stack



Source: http://wiki.ros.org/move_base

Preparing Your Sensor Data: REPs

CLEARPATH ROBOTICS™

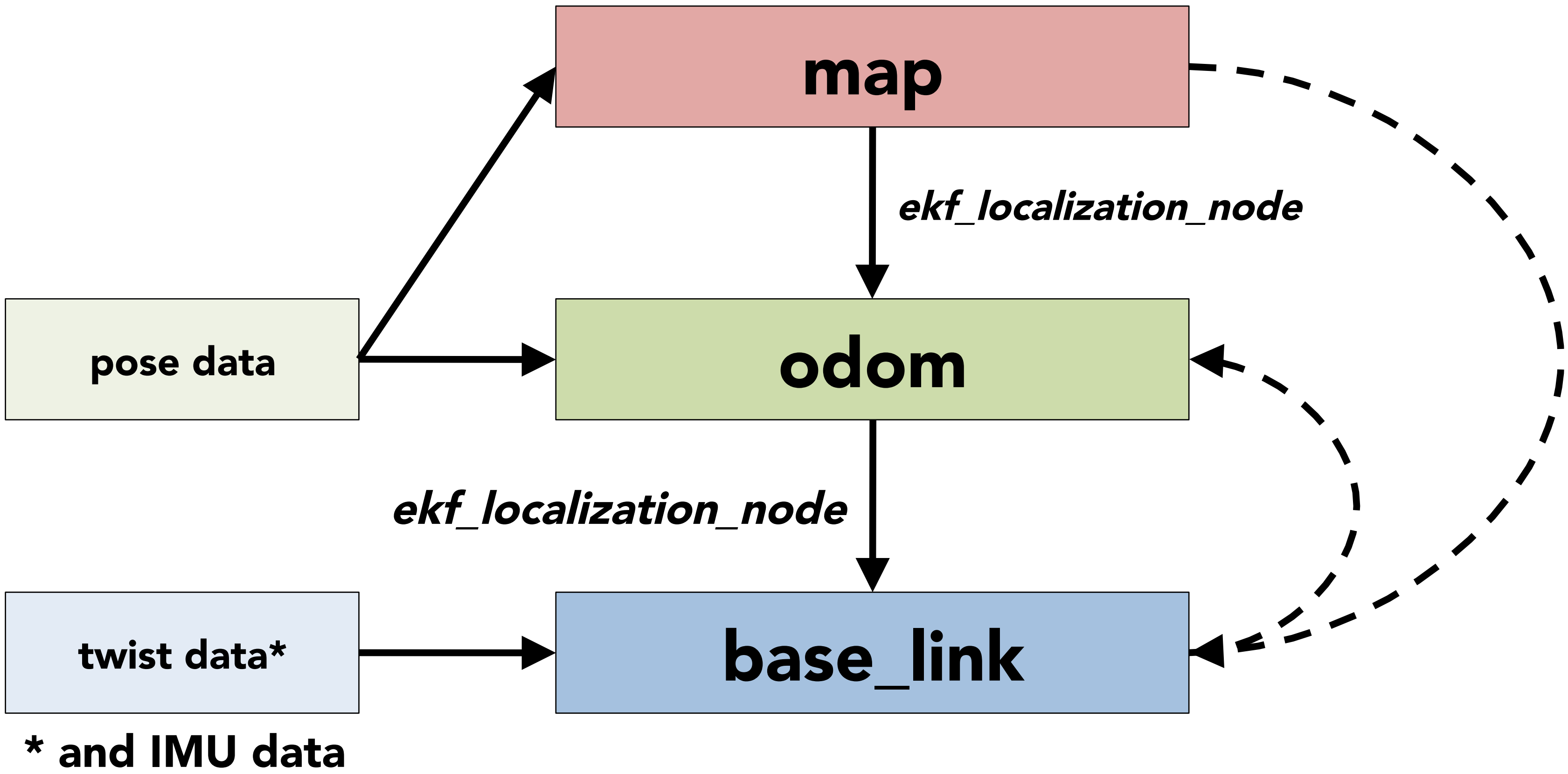
Fact: you can't spell "prepare" without REP.

Two important REPs

- **REP-103: http://www.ros.org/reps/rep-0103.html**
  - **Covers standards for units and basic coordinate frame conventions**

- **REP-105: http://www.ros.org/reps/rep-0105.html**
  - **Covers naming and semantics of the "principal" coordinate frames in ROS**

Preparing Your Sensor Data: Transforms



Preparing Your Sensor Data: Frame IDs

```
header:
  seq:   3530
  stamp:
    secs:     73
    nsecs: 724000000
  frame_id: odom
child_frame_id: base_link
pose:
  pose:
    position:
      x: 3.1032
      y: 20.164
      z: 0.0000
    orientation:
      x: 0.0000
      y: 0.0000
      z: 0.8870
      w: 0.4616
  covariance: [0.0010, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0010, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 1000000, 0.0000, 0.0000, 0.00
00, 0.0000, 0.0000, 0.0000, 1000000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 1000000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0300]
twist:
  twist:
    linear:
      x: 1.9749
      y: 0.0000
      z: 0.0000
    angular:
      x: 0.0000
      y: 0.0000
      z: 0.5737
  covariance: [0.0010, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0010, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0010, 0.0000, 0.0000, 0.000
0, 0.0000, 0.0000, 0.0000, 1000000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 1000000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0300]
```

Requires a *base_link* → *imu_link* transform

No transforms required

Preparing Your Sensor Data: Covariance

initial_estimate_covariance (P_0)

process_noise_covariance

$$\hat{\mathbf{x}}_{k|k-1} = f\left(\hat{\mathbf{x}}_{k-1|k-1}\right)$$

$$\mathbf{P}_{k|k-1} = \mathbf{J}\mathbf{P}_{k-1|k-1}\mathbf{J}^T + \mathbf{Q}$$

$$\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{H}_k^T\left(\mathbf{H}_k^T\mathbf{P}_{k|k-1}\mathbf{H}_k^T + \mathbf{R}_k\right)^{-1}$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k\left(\mathbf{z}_k - \mathbf{H}_k\hat{\mathbf{x}}_{k|k-1}\right)$$

$$\mathbf{P}_{k|k} = \left(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k\right)\mathbf{P}_{k|k-1}\left(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k\right)^T + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T$$

nav_msgs/Odometry
geometry_msgs/PoseWithCovarianceStamped
geometry_msgs/TwistWithCovarianceStamped
sensor_msgs/Imu

Configuring **kf_localization_node*

Coordinate frame specification

```
<param name="map_frame" value="map"/>
<param name="odom_frame" value="odom"/>
<param name="base_link_frame" value="base_link"/>
<param name="world_frame" value="odom"/>
```

Configuring *kf_localization_node

CLEARPATH ROBOTICS™

Input specification

```
<param name="odom0" value="/controller/odom"/>
<param name="odom1" value="/some/other/odom"/>
<param name="pose0" value="/altitude"/>
<param name="pose1" value="/some/other/pose"/>
<param name="pose2" value="/yet/another/pose"/>
<param name="twist0" value="/optical_flow"/>
<param name="imu0" value="/imu/left"/>
<param name="imu1" value="/imu/right"/>
<param name="imu2" value="/imu/front"/>
<param name="imu3" value="/imu/back"/>
```

Configuring *kf_localization_node

Basic input configuration

```
<rosparam param="odom0_config">
  [true,  true,  false,   x, y, z
   false, false, false,   roll, pitch, yaw
   false, false, false,   y velocity, y velocity, z velocity
   false, false, true,    roll velocity, pitch velocity, yaw velocity
   false, false, false]   x accel., y accel., z accel.
</rosparam>
<rosparam param="odom1_config">
  [false, false, false,
   false, false, false,
   false, false, false,
   false, false, false,
   false, false, false]
</rosparam>

<param name="odom1_differential" value="true”>
```

Configuring **kf_localization_node*

CLEARPATH ROBOTICS™

Covariance specification (P_0 and Q)

```
<rosparam param="initial_estimate_covariance">
   [0.8, 0,  ..., 1e-9]
</rosparam>

<rosparam param="process_noise_covariance">
   [0.04, 0,  ..., 0.02]
</rosparam>
```

Live Demo


I DON'T OFTEN DO LIVE DEMOS
BUT WHEN I DO, I PLAY BACK DATA FROM BAG FILES
memegenerator.net

Using *navsat_transform_node*

What does it do?

- Many robots operate outdoors and make use of GPS receivers
- Problem: getting the data into your robot's world frame
- Solution:
  - Convert GPS data to UTM coordinates
  - Use initial UTM coordinate, EKF/UKF output, and IMU to generate a (static) transform *T* from the UTM grid to your robot's world frame
  - Transform all future GPS measurements using *T*
  - Feed output back into EKF/UKF

Required Inputs

- nav_msgs/Odometry (EKF output, needed for robot's current pose)
- sensor_msgs/Imu (must have a compass, needed to determine global heading)
- sensor_msgs/NavSatFix (output from your navigation satellite device)

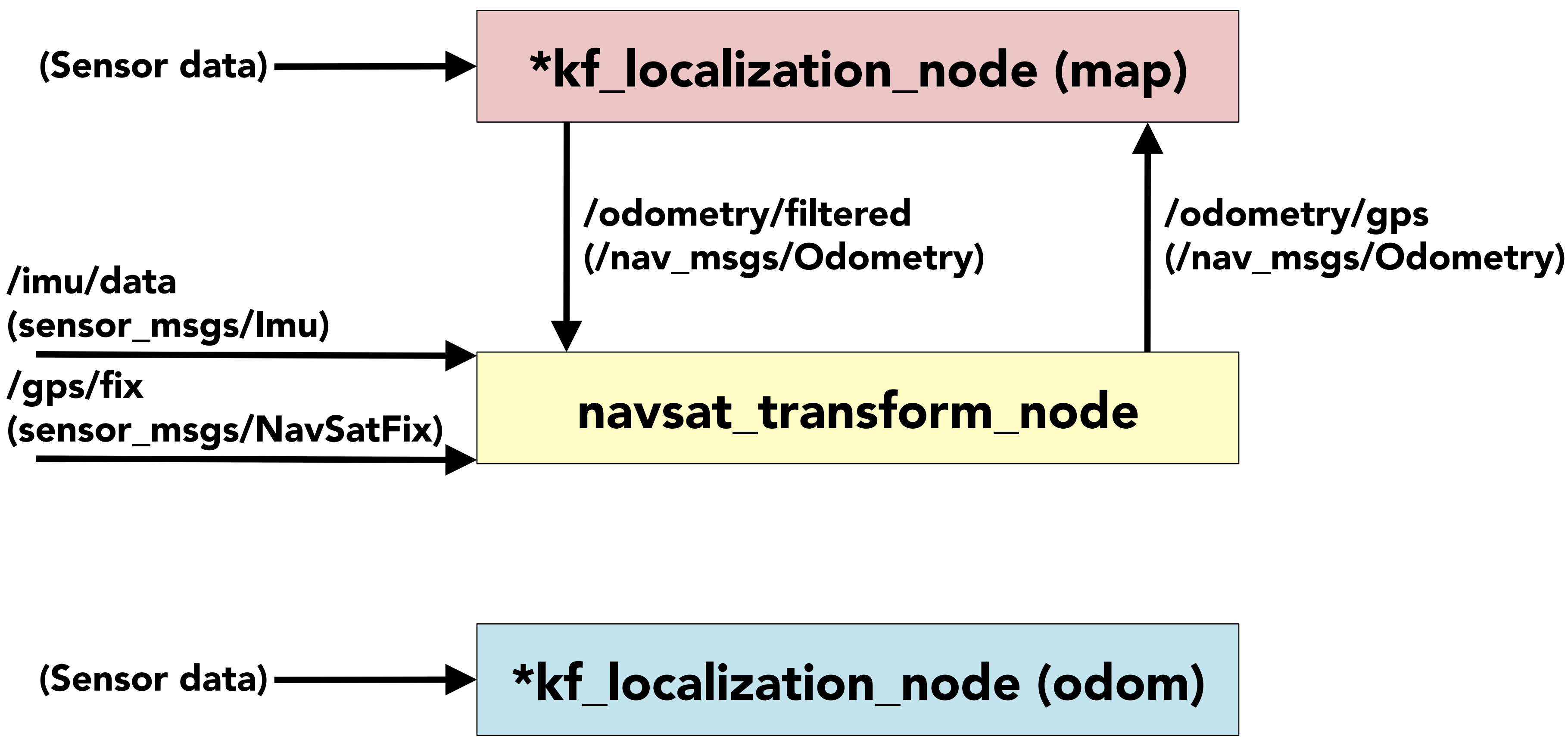
Using *navsat_transform_node*

Relevant settings

```
<param name="magnetic_declination_radians" value="0"/>
<param name="yaw_offset" value="0"/>
<param name="zero_altitude" value="true"/>
<param name="broadcast_utm_transform" value="true"/>
<param name="publish_filtered_gps" value="true"/>
```

Using *navsat_transform_node*

Typical Setup



robot_localization in the Wild

robot_localization works on a broad range of robots!

From this…
(ayrbot)

…to this
(OTTO)

Fin

CLEARPATH ROBOTICS™

Thank you!

http://wiki.ros.org/robot_localization

tmoore@clearpathrobotics.com