The Journal of Open Source Software

Tools for Prototyping with 3D Ultrasonics in ROS

Adi Singh[1], **Sebastian Dengler**[1], **and Christopher Lang**[1]

1 Toposens GmbH, Lyonel Feininger Str 28, 80807 Munich, Germany

DOI: 10.21105/joss.01531

Software

- Review
- Repository
- Archive

Submitted: 21 June 2019
Published: 23 July 2019

Summary

Ultrasound provides a cost-effective method for reliable close-range sensing. It is robust against ambient light conditions, accurately detects non-opaque surfaces, and usually has very low operational power requirements. Despite these unique advantages, ultrasonic sensing remains largely underdeveloped within the current robotics landscape. Mainstream solutions offer sensing in simple 1-dimensional contexts, like the popular HC-SR04 (Elecfreaks, n.d.), and only limited options exist for production-ready integration.

We introduce software for enabling 3-dimensional perception via Toposens (TS) sonars in Robot Operating System (ROS) (Quigley et al., 2009). This software includes a driver for handling low-level device communication, two data visualization packages, and a sync package for coordinating simultaneous operation of multiple sensors. The dynamic reconfigure library (Gassend, 2009) is used to tune signal and algorithm parameters during runtime. Compatibility with popular ROS tools like `rviz` (Kam, Lee, Park, & Kim, 2015), `pcl` (Rusu & Cousings, 2011) and `tf2` (Foote, 2013) is also demonstrated.

Ultrasonic sensing is a vital component of applications employing robotic perception, and a system can attain significantly more insights about its immediate environment by extending its ultrasonic detection to 3-D space. Toposens ROS packages are designed to help researchers quickly integrate this technology into larger robotics systems for the purposes of object avoidance, navigation, mapping and feature tracking.

Packages

The foundational component of this framework is the `toposens_driver` package, which provides an I/O bridge to any connected TS hardware. It is responsible for parsing the incoming data stream into ROS-compatible messages and maintaining a state machine for sensor configurations. The `dynamic_reconfigure` package is utilized to alter signal parameters on the run. Any changes made in the `rqt_reconfigure` GUI are received by the package and structured into firmware compatible bytes for transmission. This can assist users in finetuning a TS sensor to a specific end-user environment, which is especially helpful in prototyping applications that depend on sensor fusion techniques.

`toposens_sync` is a manager for working with multiple Toposens devices simultaneously, and it administers the lifecycle and ultrasonic emission characteristics of each sensor to coordinate incoming data-streams. This package has been designed as a thin layer handling multiple `toposens_driver::Sensor` instances that individually encapsulate all the device communication and logic flow. This architecture enables the system to cope with hardware failure, in addition to allowing each sensor's signal parameters to be changed independently without any extra configuration code.

Two visualization packages are provided to assist prototyping with TS sensors. `toposens_markers` manually builds `visualization_msgs/MarkerArray` for every scan to plot data

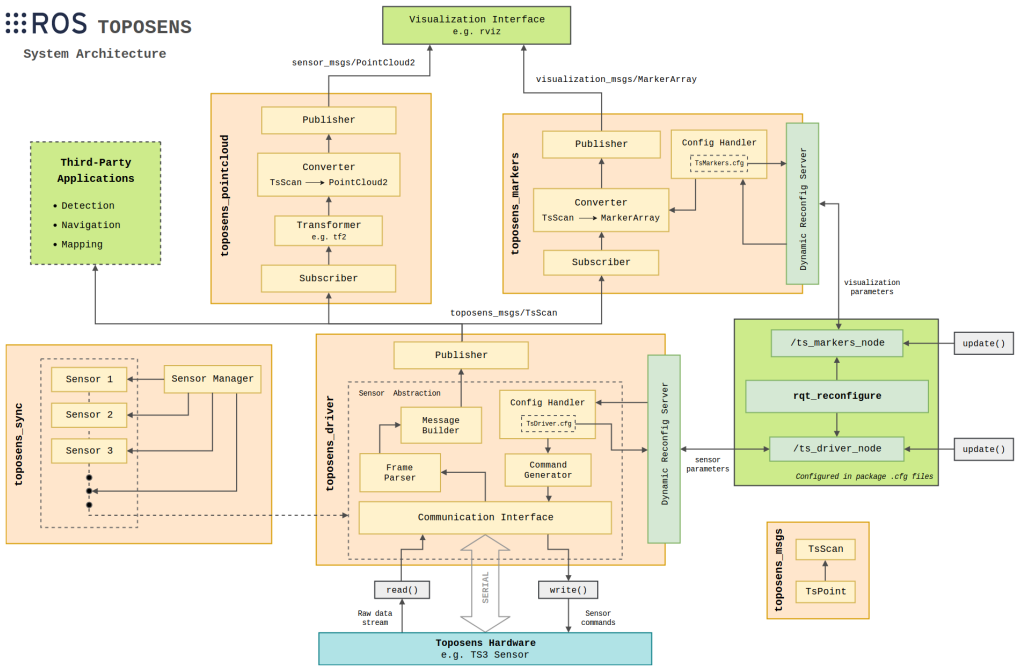

Figure 1: Overview of ROS Packages for Toposens Hardware

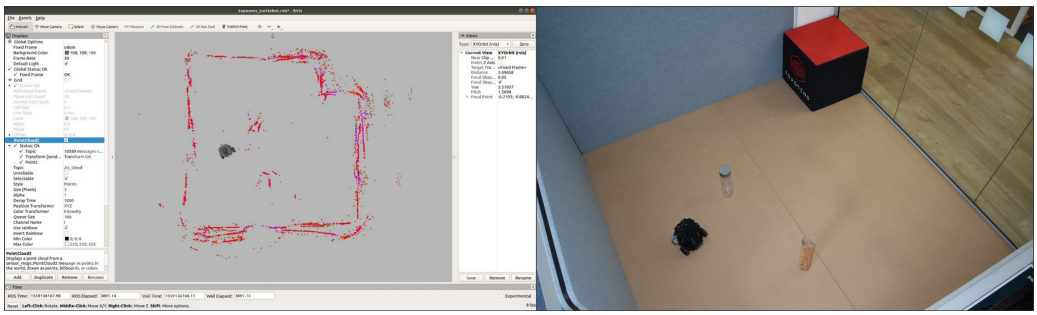

Figure 2: Data accumulated over 6 minutes from a single TS3 sensor mapping a demo arena

in a display interface like `rviz`. It comes with its own `dynamic_reconfigure` integration for managing the scale and lifetime of plotted markers. A more advanced visualization tool is found in `toposens_pointcloud`, which provides native compatibility to the Point Cloud Library (Willow Garage). This integration allows users to easily make use of other PCL modules for filtering, feature estimation and data-set registration on the sensor data. Visual attributes of the plotted data, like coloring axis, decay time and point scale, can be directly controlled from the `PointCloud2` display in `rviz`.

This package also illustrates the transformation of sensor data from its original frame using the `tf2` library; a feature that is essential for handling data collected through mobile platforms. Users can define a static transform in the launch file to enable this option, and further transformations can be applied on the data by extending the provided example.

The Turtlebot3 (Guizzo & Ackerman, 2017) map in `Figure 2` shows raw sensor output transformed to the robot's `odom` frame. No post-processing techniques for noise filtering or feature extraction have been applied at this point. Even so, one can notice obvious agreements with the real-world environment, like the recognition of arena boundaries and in-field obstacles. Of particular significance is the ample detection of the transparent wall on the right, a situation where vision-based sensing modes falter.

References

Elecfreaks. (n.d.). *HC-sr04 user guide*. https://elecfreaks.com/estore/download/EF03085-HC-SR04_Ultrasonic_Module_User_Guide.pdf.

Foote, T. (2013). tf: The transform library. In *2013 IEEE International Conference on Technologies for Practical Robot Applications (TePRA)*, Open-source computer software workshop (pp. 1–6). doi:10.1109/TePRA.2013.6556373

Gassend, B. (2009). Dynamic_reconfigure. http://wiki.ros.org/dynamic_reconfigure.

Guizzo, E., & Ackerman, E. (2017). The TurtleBot3 teacher. *IEEE Spectrum*, *54*(8), 19–20. doi:10.1109/MSPEC.2017.8000281

Kam, H. R., Lee, S.-H., Park, T., & Kim, C.-H. (2015). RViz: A toolkit for real domain data visualization. *Telecommunication Systems*, *60*(2), 337–345. doi:10.1007/s11235-015-0034-5

Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., et al. (2009). ROS: An open-source Robot Operating System. In *ICRA workshop on open source software* (Vol. 3, p. 5). http://www.willowgarage.com/papers/ros-open-source-robot-operating-system.

Rusu, R. B., & Cousings, S. (2011). 3D is here: Point Cloud Library (PCL). In *2011 IEEE International Conference on Robotics and Automation* (pp. 1–4). doi:10.1109/ICRA.2011.5980567