Resolving Message Logic Dependencies in Robotics Systems

Dr. Rahul Purandare, Dr. P. B. Sujit, Gurshabad Grover, Shivam Thukral
IIIT-Delhi

Dr. Sebastian Elbaum, Nishant Sharma
University of Nebraska, Lincoln

INTRODUCTION

Frameworks for robot software development provide a high level of abstraction for common functionalities in robots. Efficient message handling and correct communication is crucial to any such event-based system, and popular frameworks like the Robot Operating System[1], and MOOS-IvP[2] provide a model which consists of publishers (message sources) and subscribers (message sinks) to facilitate this communication.

Messages in ROS are specified using a descriptive language.

```
int32 xPosition
int32 yPosition
string name
```

Fig. i – Example message file

Our work consists of an analysis of bug reports we studied, and solution to some of the problems we encountered.

PRUNING FOR BLOATED MESSAGES

Fields in a message may be unused by the subscriber. These fields, having no utility of their own, take up additional and unnecessary space in each transmission.

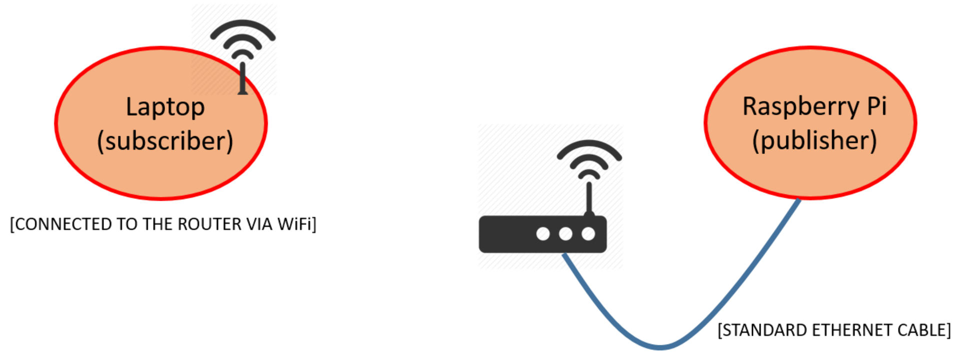

Fig. ii – Our setup

We want to investigate how heavier messages affect performance of a robot. We chose the following metrics:

- Average latency for transmission
- Percentage of messages dropped

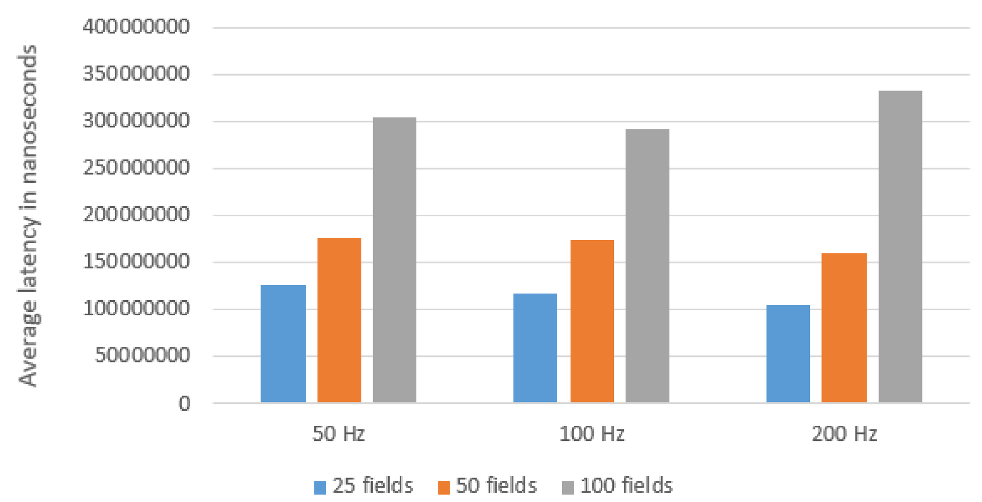

Fig. iii – Average latency in message transmission

The results confirm that average latency in message transmission and percentage of messages dropped increase with an increase in message size.

Fig. iv – Percentage of messages dropped

Proposed solution: Pruning bloated messages

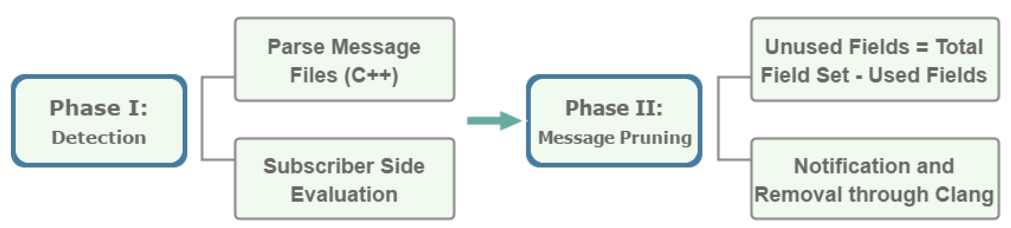

Fig. v – Multi-stage analysis for message pruning

INCONSISTENCY BUGS

Message fields are filled by a publisher and processed for information by a subscriber. For any given field in a message, a publisher may fill it from a set of valid values. This set of values may not be consistent with what the subscriber is expecting.

This can lead to several problems such as the subscriber ignoring a syntactically valid message, or even a malfunction or crash. We collected bug reports exhibiting such issues.

Using symbolic execution (with KLEE), we are developing a tool which automates the process of ensuring consistency.

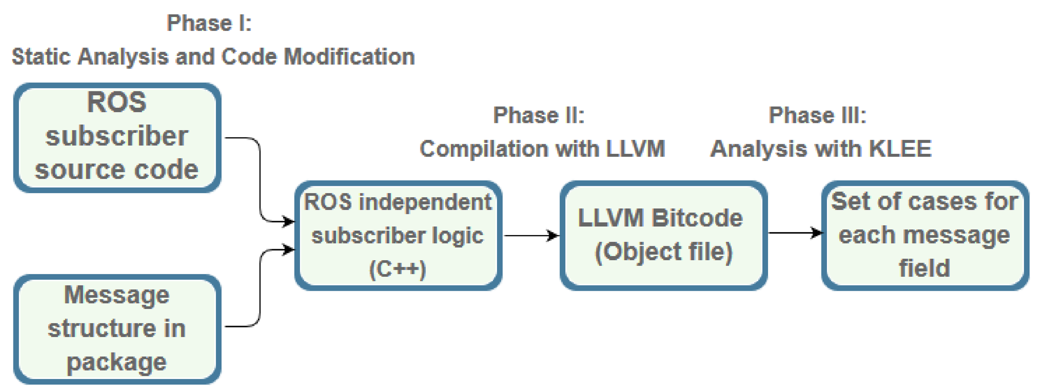

Fig. vi – Solution workflow for detecting inconsistencies

UPDATING OF ROS LIBRARIES

Since a robotics project can be maintained and worked on by several developers, it is highly likely that some developers have different versions of these libraries.

Depending on the type and scale of change in the libraries, there might be functions which return unexpected values. This can cause unexpected behaviour in the robotic system.

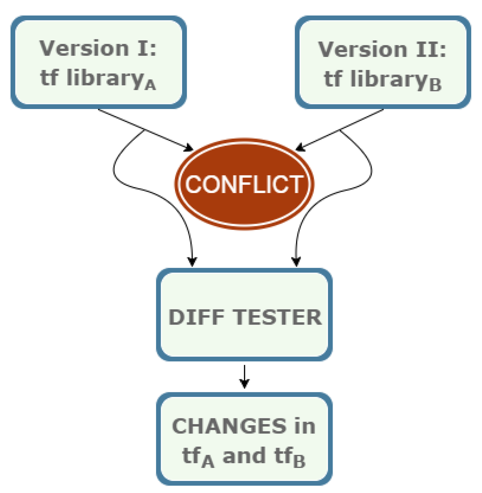

Fig. vii – Flow to detect changes in libraries

CHANGE IN MESSAGE PUBLISHING LOGIC

All components in the system have the ability to publish or subscribe to any channel. Thus, identifying components which depend on a specific component is not trivial.

Continuing the work started in *Extracting Conditional Component Dependence For Distributed Robotic Systems*[7], we wanted to modify a tool which compares two versions, and detects the conditions under which a message is published.

We added:

- Automation of comparison over versions
- Support for `for`, `while`, `switch` statements
- Automation of all phases
- Build process detection

CONCLUSIONS

Using a variety of techniques like static program analysis and symbolic execution; and our knowledge of how these robot frameworks work, we have implemented ways in which the performance of the system and the pace of development in robotics can be increased.

REFERENCES

[1] ROS: http://www.ros.org/
[2] MOOS: http://www.robots.ox.ac.uk/~mobile/MOOS/
[3] Understanding ROS Nodes: http://wiki.ros.org/ROS/Tutorials/UnderstandingNods
[4] rxgraph: http://wiki.ros.org/rxgraph
[5] KLEE: http://klee.github.io/
[6] Cristian Cadar, Daniel Dunbar, Dawson Engler: KLEE: Unassisted and Automatic Generation of High-Coverage Tests for Complex Systems Programs (OSDI 2008)
[7] R. Purandare, J. Darsie, S. Elbaum, M. Dwyer: Extracting conditional component dependence for distributed robotic systems (2012 IEEE/RSJ International Conference on Intelligent Robots and Systems)