User Manual

Introduction to panda robot fault injection framework

This project is a part of my Master Thesis at University of Stuttgart.

This source code works as a environment playground for franka emika panda robot only, following steps depict quick start guide to setup the environment and run fault injection experiments.

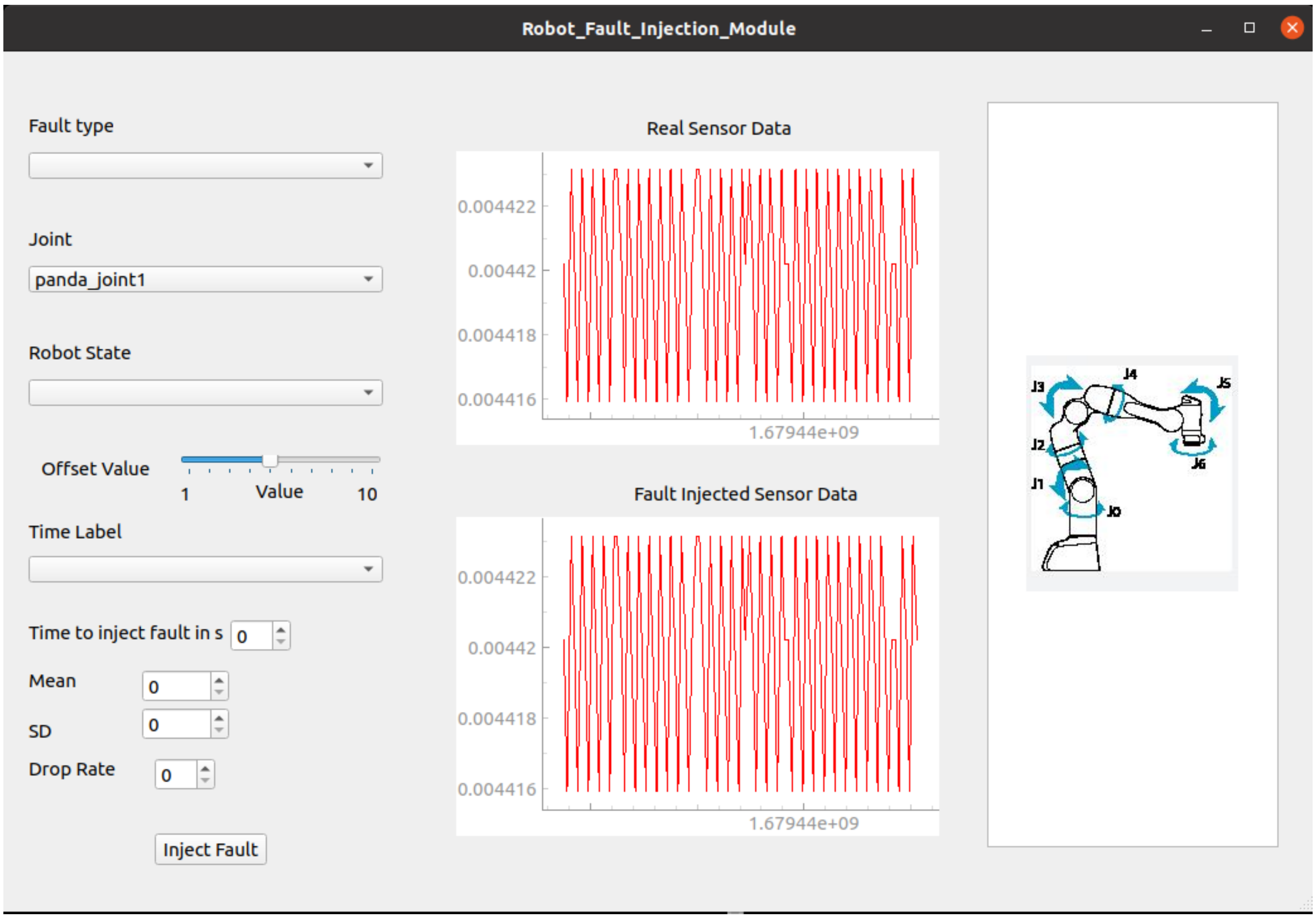
If you want to use Robot FI Tool to inject faults into different robot please follow this repository

Package Structure

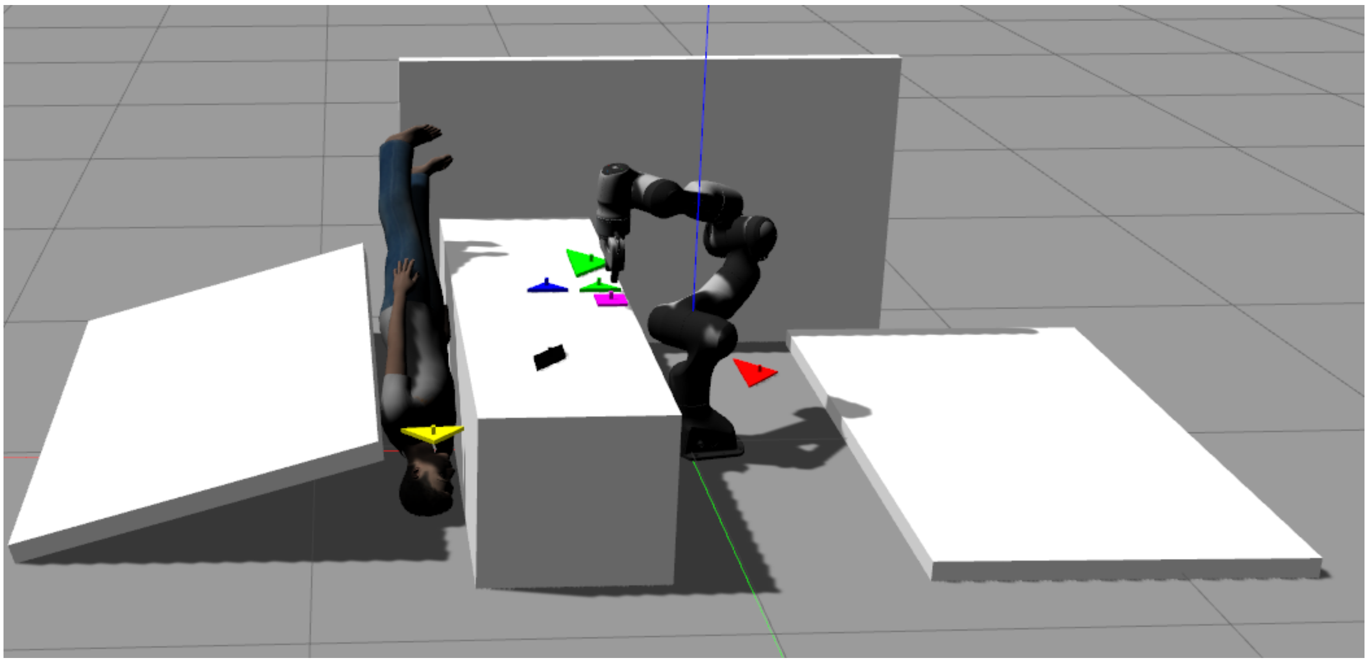
this section briefly explains the structure of the package and explains customizationa options and some important folders.

- data_vx: stores data for fault injection experiment
- devel: contains all the ROS build files
- install: ROS install directory
- plots_vx: stores plots generated fromm fault injection experiment
- src
  - moveit tutorial: moveit controller code and configurations for panda robot
    - custom pick place code (editable version) please go through the ROS C++ design schematic before editing and building this file : pick_place_fib.cpp
  - panda gazebo: panda robot package to support gazebo simulation this package also houses seven piece puzzle environment
    - resources: has all the required models and 7pp sdf for simulation
      - models: add or remove models if wanted
      - world: defines gazebo world the main world file used is tangram.world
  - robot_fi_tool: this package containes the fault injection module
    - config: configuration files
    - launch: launch files
    - msg: custom message
    - src: source code
    - utils: code for FI experiment

Frontend



Scenario



Quick Start

Requirements

- `sudo apt-get install qt5-default`
- `pip3 install -r requirements.txt`

build

`catkin build -j4 -DCMAKE_BUILD_TYPE=Release -DFranka_DIR:PATH=~/libfranka/build`

Start simulation

- `source devel/setup.bash`
- `roslaunch panda_gazebo put_robot_in_world.launch`

Start FI Module and person sim

- `roslaunch robot_fi_tool fault_module.launch`
- to inject fault using GUI: `rosrun robot_fi_module fib_gui_v2.py`

or

Start random FI Process

- random planning and execution: `roslaunch robot_fi_tool rand_fault_injector.launch`
- random real-time: `roslaunch robot_fi_tool real_time_rand_fault_injection.launch`

FI Experiment

- `bash pipiline.sh`

this starts random fault injection experiment and saves the bag files into data_v4

Docker setup

The package is also supported with docker, we provide a docker image here, it is built with noetic ros docker image and this section shows how to setup the image and start the container, and run the graphics to support gazebo and rviz

- `docker pull sharathnataraj1304/panda_robot_fault_injection_framework:latest`

- `sudo xhost +si:localuser:root`

- `docker run --gpus all -e DISPLAY=$DISPLAY --env NVIDIA_VISIBLE_DEVICES=all --env NVIDIA_DRIVER_CAPABILITIES=all --env DISPLAY --env QT_X11_NO_MITSHM=1 -v /tmp/.X11-unix:/tmp/.X11-unix -v /dev:/dev --net=host --privileged -it sharathnataraj1304/panda_robot_fault_injection_framework:latest /bin/bash`

- `source /opt/ros/noetic/setup.bash`

- in the other terminal start roscore locally `source /opt/ros/noetic/setup.bash` and `roscore`

- test the topics within the docker - `rostopic list`

- `cd panda_fault_injection_block`

- `rm -r build/` and `rm -r devel/`

- `catkin build -j4 -DCMAKE_BUILD_TYPE=Release -DFranka_DIR:PATH=~/libfranka/build`
- `source devel/setup.bash`
- `bash docker_exec.sh`

Introduction to general robot fault injection framework

Robot FI Tool to inject faults into any robot please follow this repository

Package Structure

this section briefly explains the structure of the package and explains customizationa options and some important folders.

- src
  - robot_fi_tool: this package containes the fault injection module
    - config: configuration files
    - launch: launch files
    - msg: custom message
    - src: source code
    - utils: code for FI experiment

Quick Start

Requirements

- `sudo apt-get install qt5-default`
- `pip3 install -r requirements.txt`

Install

- add required robot modules and its ROS support packages

Controller setup

we can use a moveit C++ robot controller this guide shows how to setup any robot controller to work with robot fault injector pakcage

- Create publisher and subsriber use the same topic as shown in the below figure.

Required publisher and subscriber

- publish iterations: since the robot perform multiple iterations of a similar movement

Coordinates

- publish status message and state of the robot state here means the pose of the robot

message_publisher

- sleep for 5 seconds befor starting the moveit call

sleep

start the simulation

once the robot controller is setup launch the robot simulation.

build the robot simulation package and robot fi module

`catkin build -j4 -DCMAKE_BUILD_TYPE=Release -DFranka_DIR:PATH=~/libfranka/build`

Start simulation

- `source devel/setup.bash`
- `roslaunch panda_gazebo put_robot_in_world.launch`

Start FI Module and person sim

- `roslaunch robot_fi_tool fault_module.launch`
- to inject fault using GUI: `rosrun robot_fi_module fib_gui_v2.py`

or

Start random FI Process

- random planning and execution: `roslaunch robot_fi_tool rand_fault_injector.launch`
- random real-time: `roslaunch robot_fi_tool real_time_rand_fault_injection.launch`

FI Experiment

- `bash pipiline.sh`

this starts random fault injection experiment and saves the bag files into data_v4