CALIFORNIA SPACE GRANT CONSORTIUM

National Aeronautics and Space Administration

NASA Artemis Student Challenge Opportunities

LEAPFROG

Software Challenge Informational Session #2 for Competition Teams

Student Team

UC San Diego

USC University of Southern California

Berkeley UNIVERSITY OF CALIFORNIA

Faculty Team

John Kosmatka (PI)
University of California, San Diego

David Barnhart (Co-PI)
University of Southern California

Dan Zevin
University of California Berkeley

April 30th, 2021

Reminder:
LEAPFROG is 1 of 3 Theme-3 2020 Awards

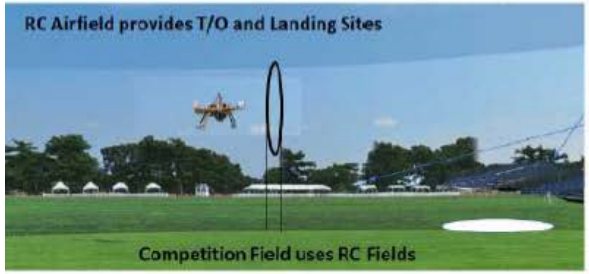
THEME 3 AWARDS – PILOT ARTEMIS STUDENT CHALLENGES

- Lunar/Martian Lander skills competition using existing technology to execute the competition in Earth's gravity and atmosphere
- Develops Artemis-relevant systems engineering and integration skills, and requires innovations to perform well against the other teams
- Requires the development and evolution of critical thinking and hands-on skill sets to enable humankind's next great steps off of Earth

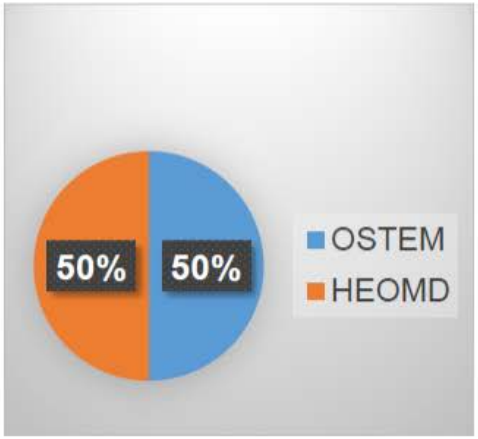
CALIFORNIA

CALIFORNIA SPACE GRANT CONSORTIUM

Possible Competition Metrics:
- Flight Time
- Stability and Hold
- Navigation Precision
- Fuel Useage
- Multi-Hop Translation
- Various Guidance Challenges
- Etc



RC Field supports Audience Member Viewing Stands for public involvement



Artemis LEAPFROG Project Goals

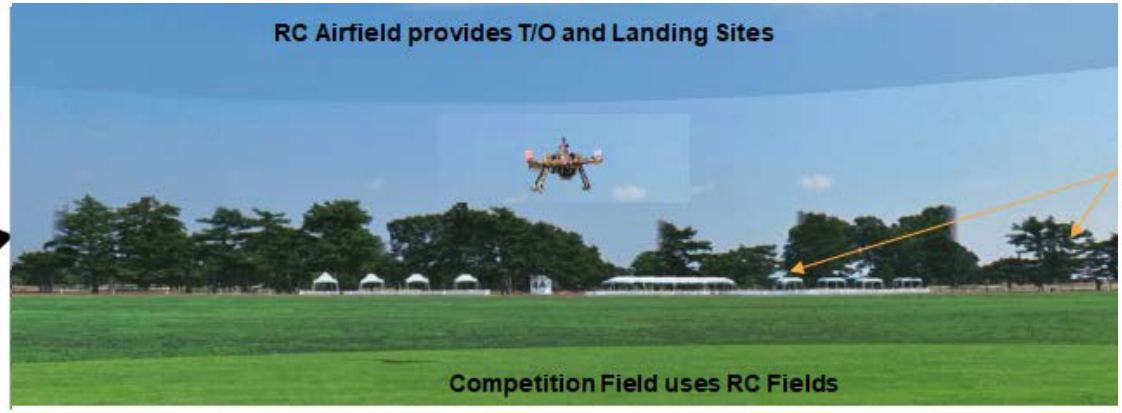
- Update existing USC LEAPFROG lander prototype
- Develop competitions around the Lunar Lander prototype
- Run Competitions in Summer of 2021!

LEAPFROG Operations Center:
Up to 500 meter range

Possible Competition Metrics:
- Flight Time
- Stability and Hold
- Navigation Precision
- Fuel Useage
- Multi-Hop Translation
- Various Guidance Challenges
- Etc

Wireless signal booster

Laptop Control



RC Field supports Audience Member Viewing Stands for public involvement

RC Airplane Pits: LEAPFROG Fueling and Pre-Flight C/O

LEAPFROG Sets up PostFlight Assessment and Scoring

Run a national competition in 2021!

Updated Timeline

Activities	Start Date	End Date
Announce S/W Competition	3/4/21	3/5/21
Run Signups for all potential Teams	3/5/21	**5/7/21**
Run PX4 and Simulation Webinars	3/28/21	5/12/21
Finalize Teams, Notify Acceptance	**5/12/21**	**5/17/21**
Seminar #1	***S/W Tools***	***4/7/2021***
Seminar #2	***Running the Sim***	***4/30/2021***
Seminar #3	**Competition Metrics**	**5/12/2021**
Run Software Challenge (4 Weeks)	**5/24/21**	**6/18/21**
Finalize and Contact Winners	**6/21/21**	6/30/21
Ship out LEAPFROG Vehicle Kits	7/1/21	7/15/21
Run Vehicle Building Bootcamps & Run Exhibition Flights	**7/28/21**	**8/11/21**
Submit Final Report to NASA	8/5/21	8/31/21

Update on Initial Team's (as of 4/30/2021)

Current Teams identified in our Registration Site:

Idaho State University
Purdue University
New Mexico Tech
City College of San Francisco
New Mexico Highlands University
California State Polytechnic University, Pomona
University of Illinois Urbana-Champaign
UC Berkeley
University of West Florida
Arizona State University
University of Texas at Austin
Texas A&M University
New Mexico State University

Notes:

- **Only US universities or institutions can sign up for possible Lander Kits**
- **International participation is possible as part of a USC team**
- **Every Team MUST have a Faculty Mentor signed up**

Software Competition Challenge and Exhibition Flights

~ March to June 2021

Virtual Software Challenge

Crater Landing

Goal: Locate & land in center of designated crater

Obstacle Avoidance

Goal: Identify & maneuver past potential hazards

- Virtual S/W Challenge run remotely
- Skills enable transfer between Simulation and Flight
  - Waypoint navigation, control methodology, obstacle avoidance
- Two Challenges earn Points, highest point earner wins
- Top 6 teams will advance to Hardware Kit Build and Top 3 for Flight Exhibition
- Possible that more kits may be available to competition teams (US only)

~ June to August 2021

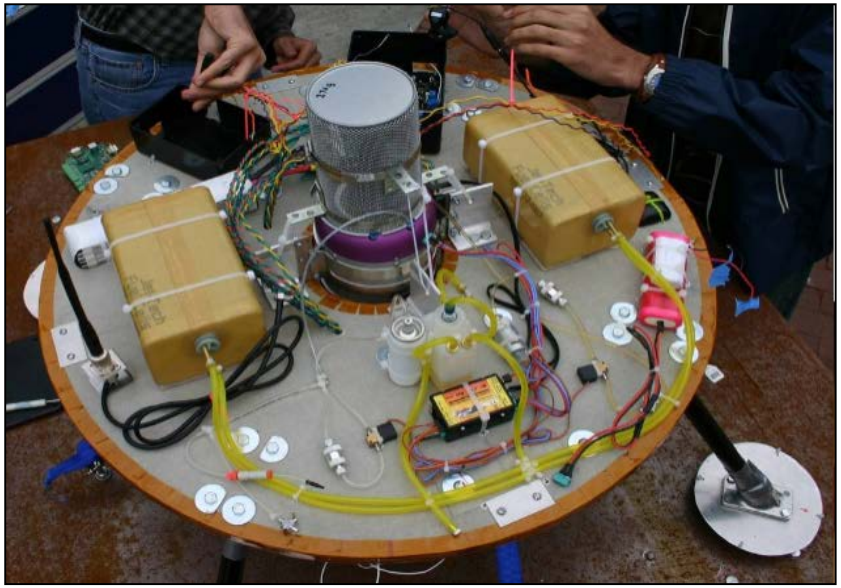
Kit Boot Camp and Flight Exhibition

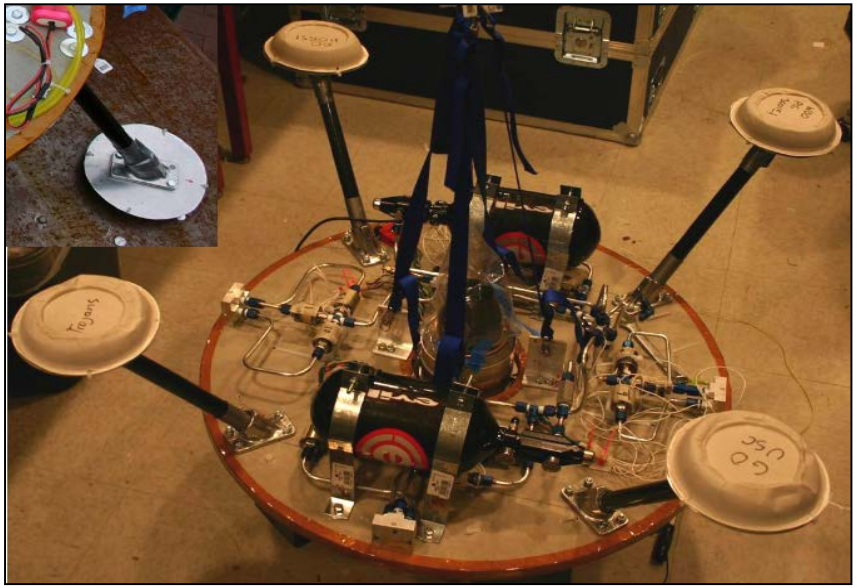
Goal: Enable teams from S/W challenge to build full kit (without Engine), and run Exhibition Flights with Winning S/W

- Teams from S/W Challenge (first 6 at minimum) will receive full hardware "Kit"
- Boot Camp will be run during summer to help all teams "Build" their Lunar Lander Vehicle
- Exhibition Flights will be run in CA for the Top three Teams in the S/W challenge

What is LEAPFROG?

LEAPFROG (**Lunar Entry and Approach Platform For Research On Ground**) was started as a multi-semester design-to-flight student hands-on training activity through Astronautical Engineering and Information Sciences Institute at the University of Southern California in **2006**.

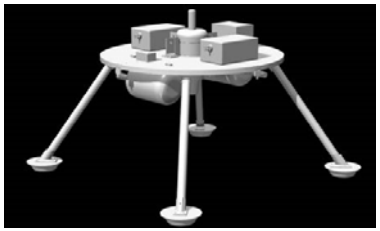




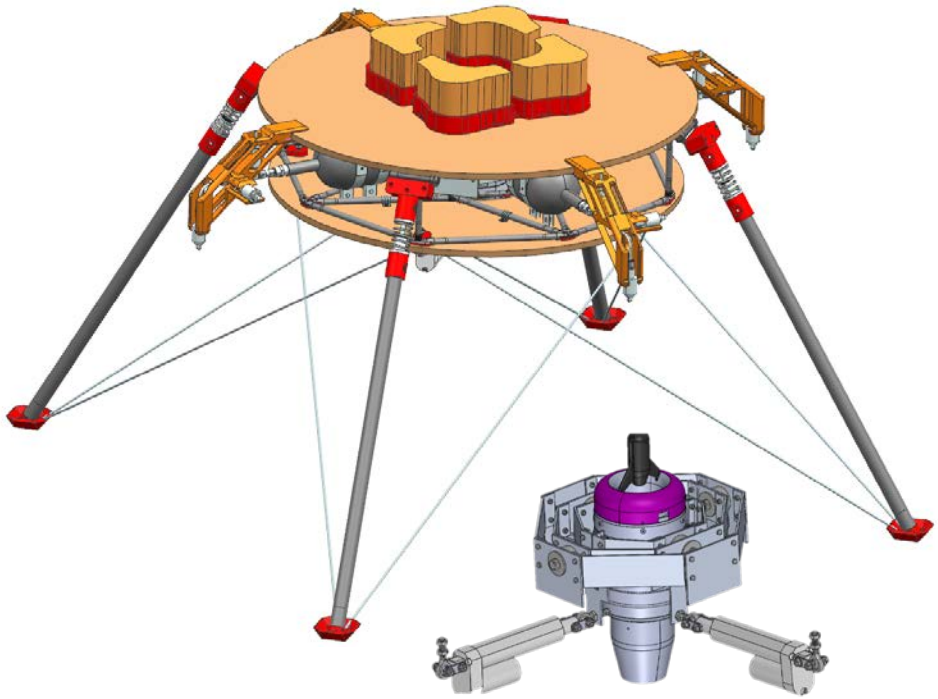
LEAPFROG

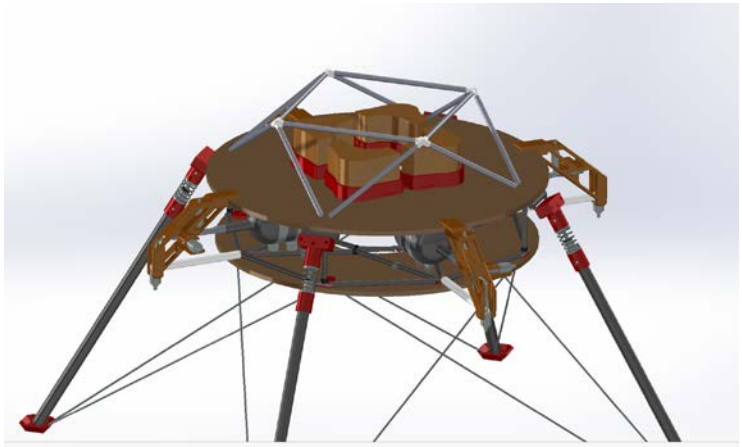
LEAPFROG taught Lander Systems Engineering through Hands-On Flight Test

LEAPFROG Generational Vehicles designed to fly multiple times in single day



Artemis LEAPFROG Challenge Vehicle has been modified for Safety and Ease of Ops





Gen-2 LEAPFROG System Features:

- Gimbaled JetCat P300 Jet Engine for translation
- Cold gas ACS w/4 independent paintball bottles, fillable anywhere
- Carbon fiber chassis built with mechanical fuses and spring joints for hard landing and failsafe landing structure
  - "Roll cage" on top and bottom
  - Designed to land on Slopes
- Onboard GNC Avionics w/ GPS fence
- Two low Cost foam core fiberglass platforms, lightweight and strong, replaced easily, more surface area for mounting
- 4 separate small Kerosene tanks incorporated onto centerpoint, minimizes slosh, spreads CG

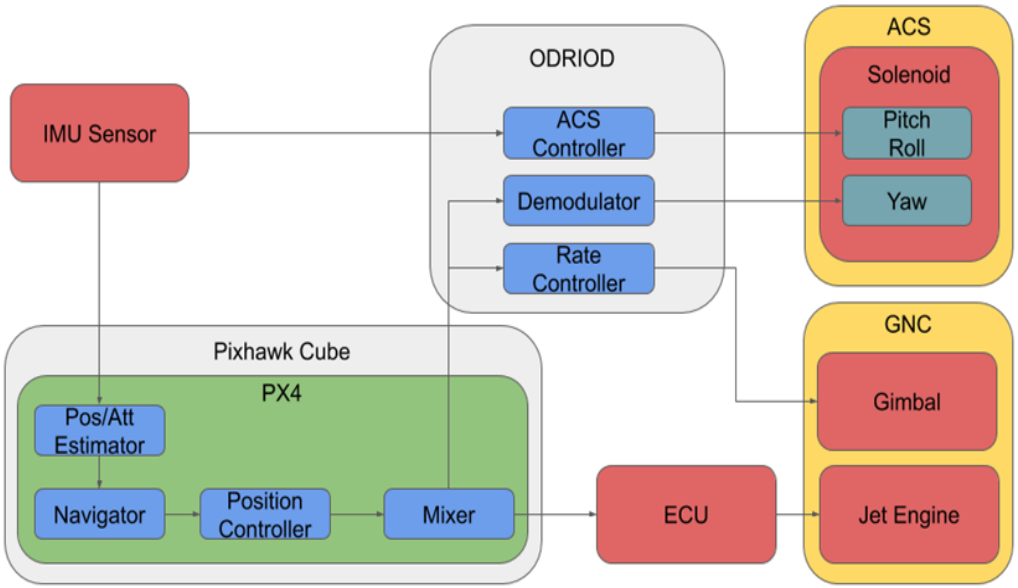
Update on Flight Vehicle Development and Test

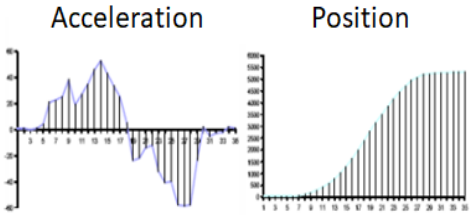
- First integrated wireless control and startup/idle of Jet Engine
- Validated ACS valve operation during engine operations
- Validated initial wireless data acquisition

Modifications to configuration and electronics from first test in work!

Software Competition exercises Lunar Lander based Guidance, Navigation and Control

- LEAPFROG Gimbal control is tied into a COTS Flight controller, **PixHawk**
  - Open source software PX4 on low cost controller, able to be used by all University teams across country
  - The “code” challenge will test programming in control methods for relative positioning, altitude control, waypoint navigation, with sensor limits on velocity and angle



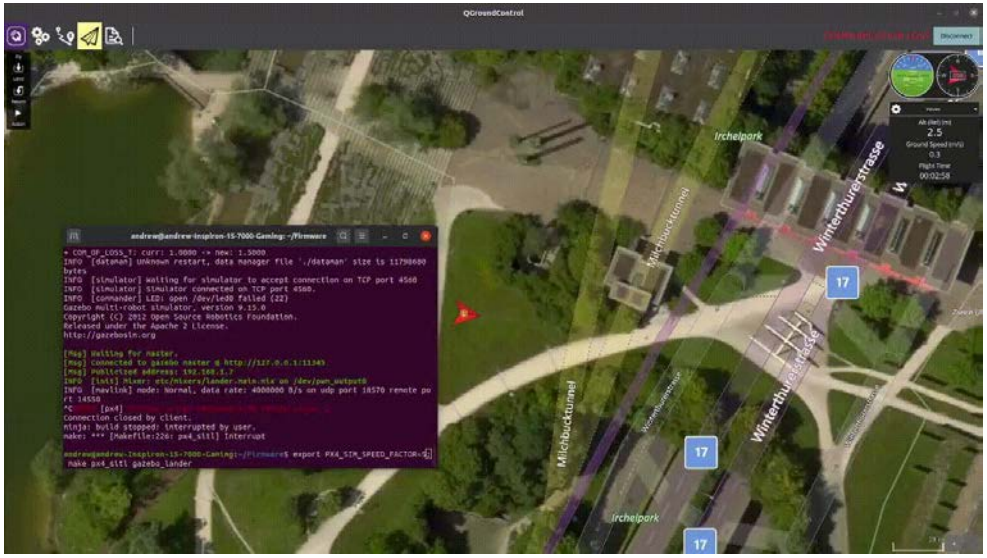


Position Estimation Algorithm Graph
(diagrams from paper on IMU sensor)

Image provided by PX4 documentation

Software Challenge uses PixHawk Simulation Software

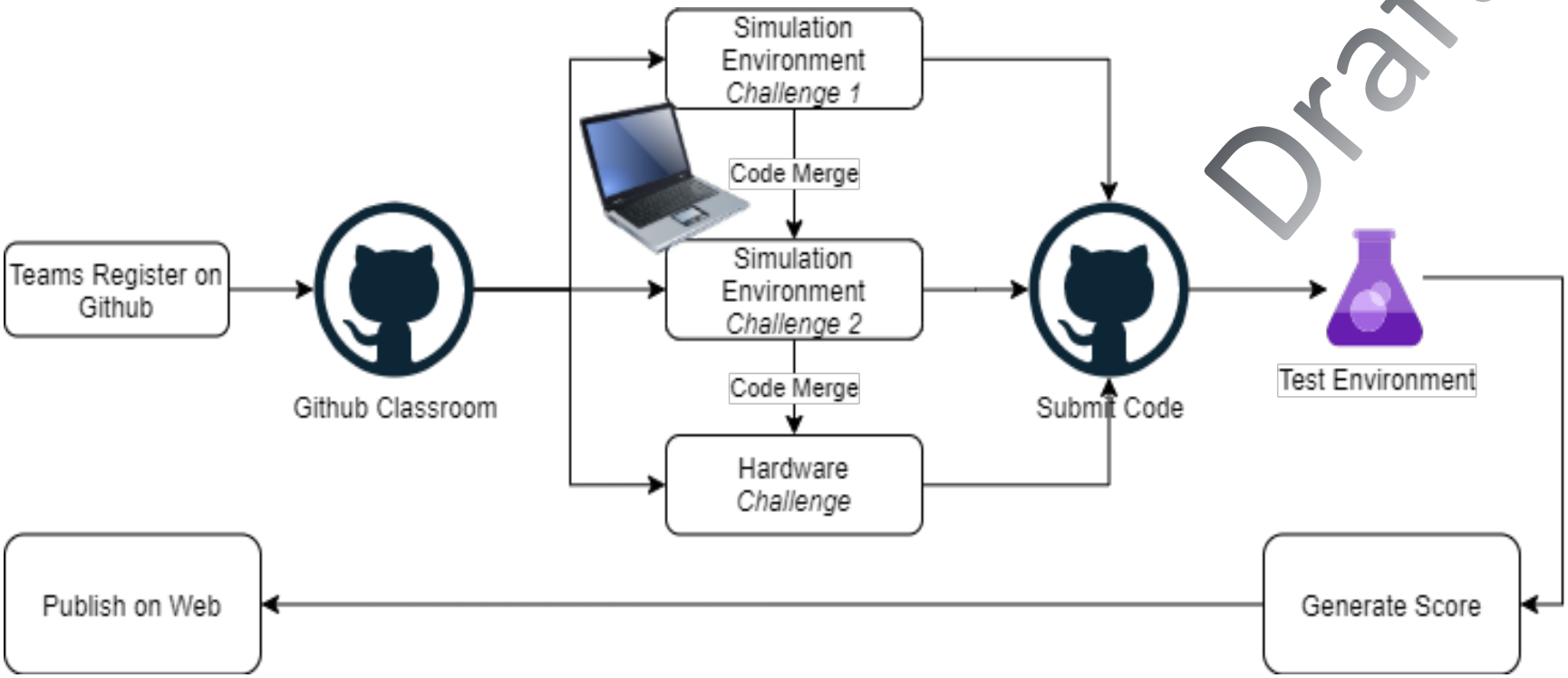
- Basic simulation with plant model of LEAPFROG components created
- Generic Earth centric physics model applied, uses Gazebo for visualization
- PixHawk controller algorithms integrated with open source tools into simulation
- Scenario provided to mimic dimensions and geography of competition RC Field



- *The bridge between the Software and Flight Challenge is the PIXHAWK CONTROLLER*

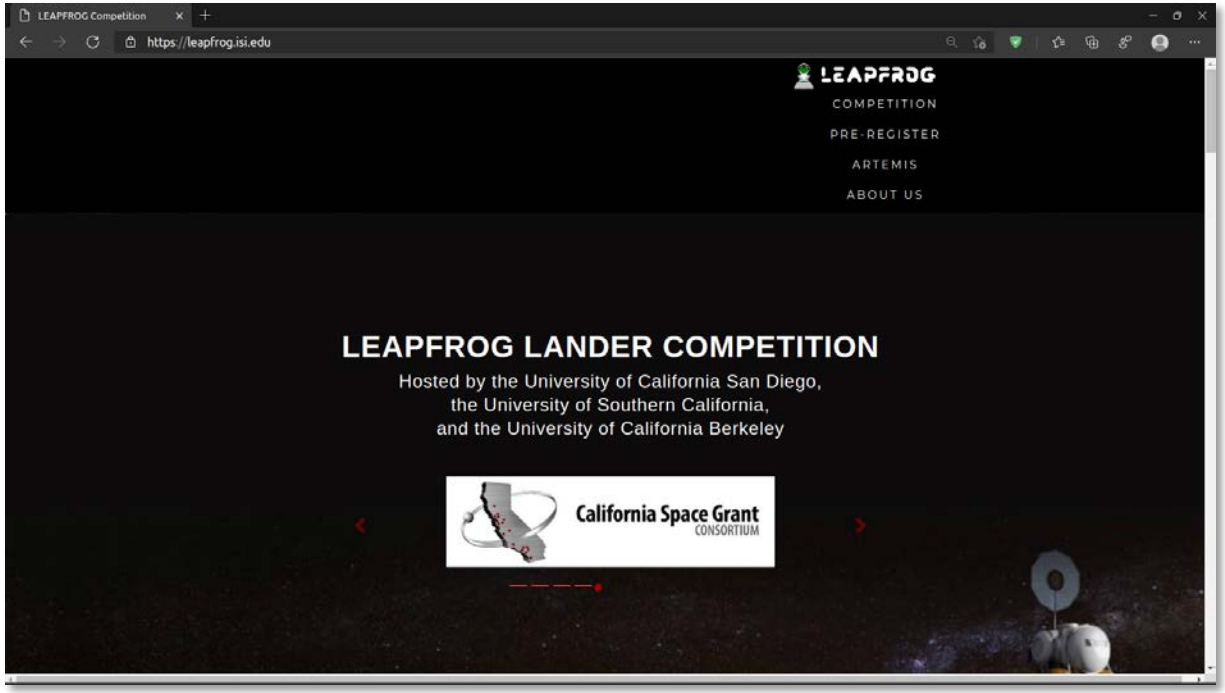
Initial Plan for Teams to share Simulation Code

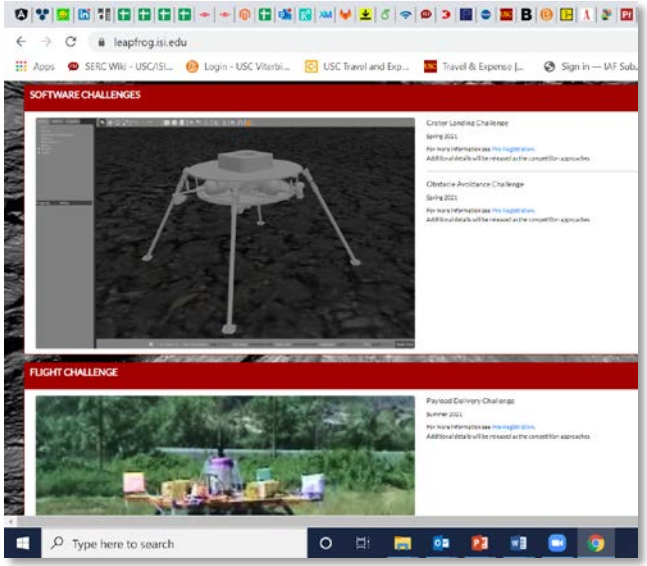
(Presented in Seminar #2 and #3 and will be put on the Website)



Where to get more Information?

- **“LEAPFROG.ISI.EDU”**

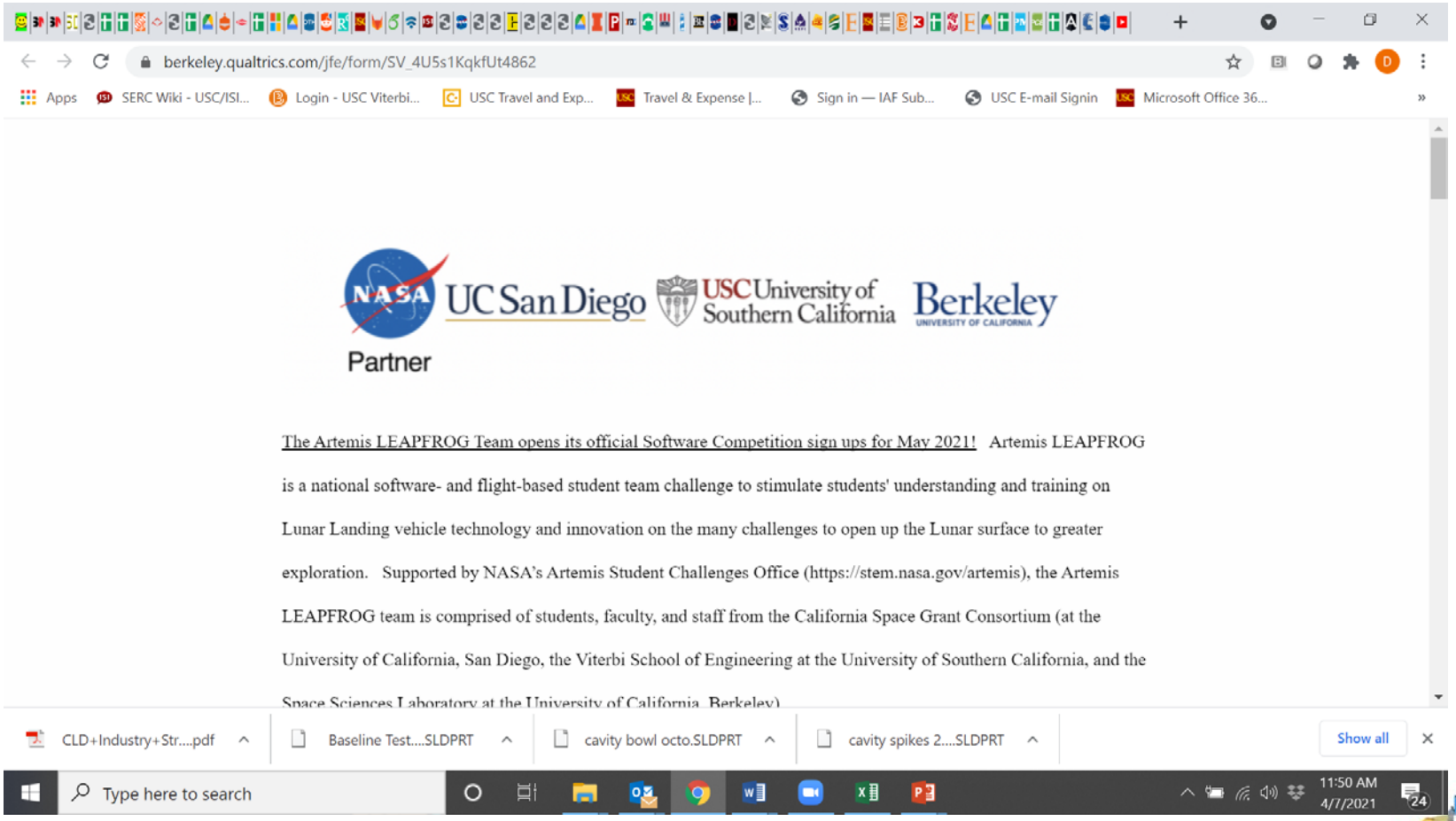




- **Competition Information will be provided**
- **Vehicle build updates**
- **Link to the Registration**
- **Links to NASA Artemis STEM Project Information**
- **Information about the LEAPFROG Team members**

Registration Page is LIVE, SIGNUP BEFORE MAY 7th!

- Dedicated Survey Link located on leapfrog.isi.edu web page
  - ***https://berkeley.qualtrics.com/jfe/form/SV_4U5s1KqkfUt4862***



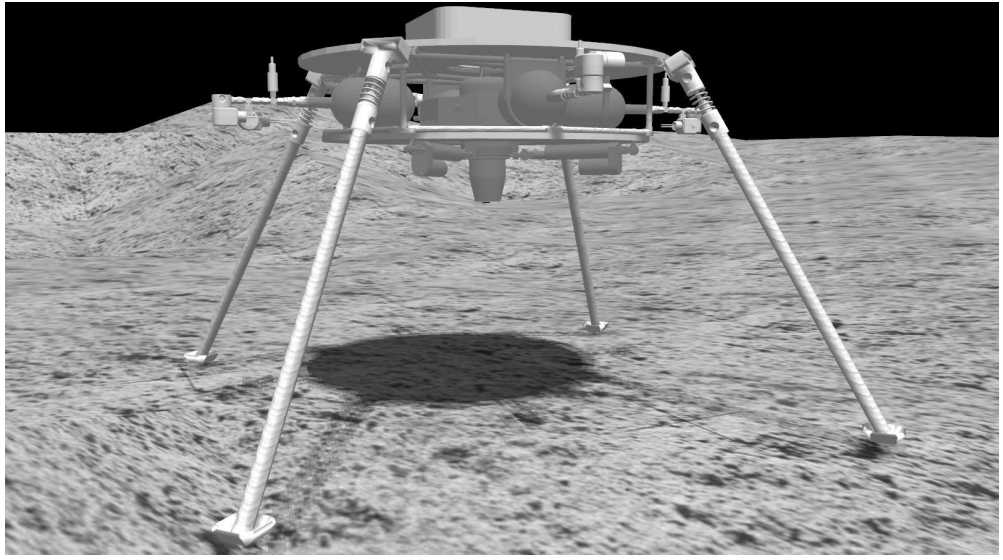
Whom to Contact

- **Fill out registration on Website (leapfrog.isi.edu)**
- **Information E-mail: leapfrog@usc.edu**
- Technical and Competition Contact:
  - Prof David Barnhart, Barnhart@isi.edu
- Programmatic Contact:
  - Prof John Kosmatka, jkosmatka@ucsd.edu

DETAILS FOR SOFTWARE INSTALLATION AND FIRST RUN!

LEAPFROG Webinar 2: Setup and Usage

April 30, 2021



Today's Overview

1. **Walk through installation**
2. **Demo Usage**
3. **Configurations**

Installation: System Requirements

Recommended OS: Ubuntu 20 (or variant)

- You can technically run PX4, Gazebo, and ROS on OSX or Windows, but we won't be able to support this.
  - Windows with Windows Subsystem for Linux (WSL) will have the greatest likelihood of working if Ubuntu is not an option.
- For teams unable to set up a Linux system, using a VM is recommended. https://www.vmware.com/

Basic essential system packages:

```
sudo apt-get install git g++ cmake curl make python3-pip -y
```

Installation: Gazebo

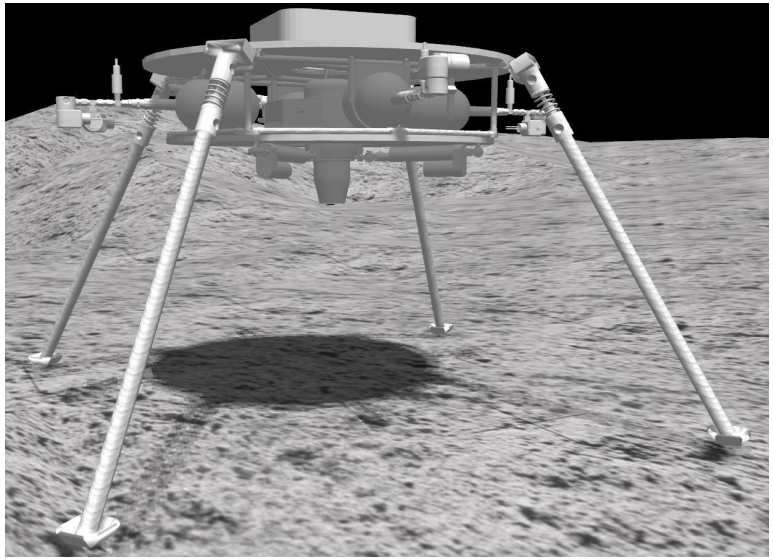
With Ubuntu 20, Gazebo 11 should be used.

Default installation script:

```
curl -sSL http://get.gazebosim.org | sh
```

Notes:

- Detailed instructions on how to install Gazebo on Ubuntu/Debian based systems can be found here: http://gazebosim.org/tutorials?tut=install_ubuntu&cat=install
- Github Repo: https://github.com/osrf/gazebo



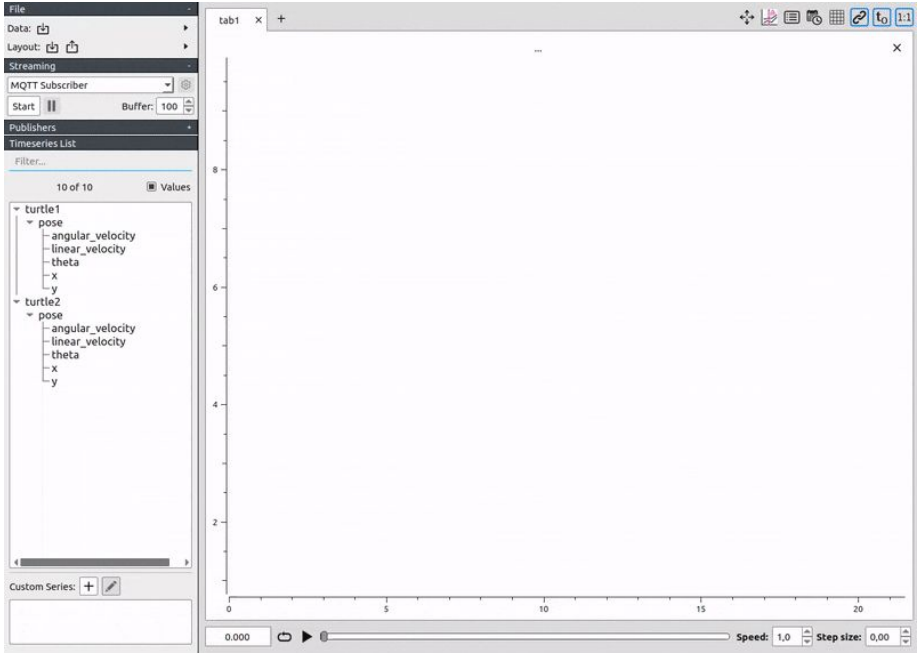
Installation: PlotJuggler

Recommended snap installation:

sudo snap install plotjuggler

References:

- Github repo for reference: https://github.com/facontidavide/PlotJuggler
- Source for snap: https://snapcraft.io/plotjuggler
- ROS wrapper/package: http://wiki.ros.org/plotjuggler
- Official website: https://plotjuggler.io/



Installation: QGroundControl

1. Setup

   sudo usermod -a -G dialout $USER

   sudo apt-get remove modemmanager -y

   sudo apt install gstreamer1.0-plugins-bad gstreamer1.0-libav gstreamer1.0-gl -y

2. Download QGroundControl AppImage:

   curl -O https://s3-us-west-2.amazonaws.com/qgroundcontrol/latest/QGroundControl.AppImage

3. Change permissions of AppImage

   chmod +x ./QGroundControl.AppImage

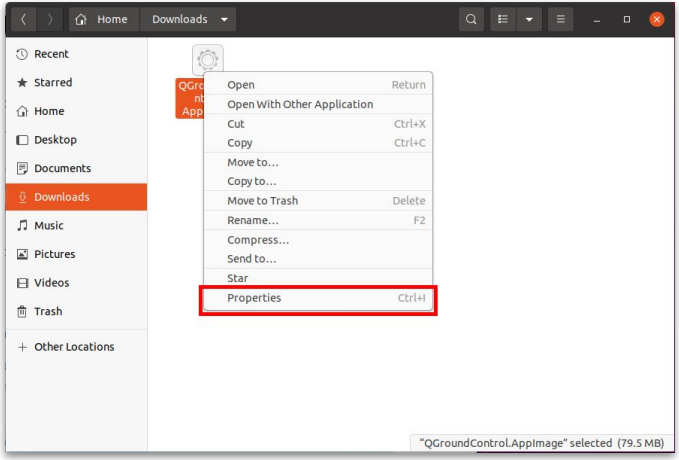
4. Run AppImage

   ./QGroundControl.AppImage (or double click actual file)

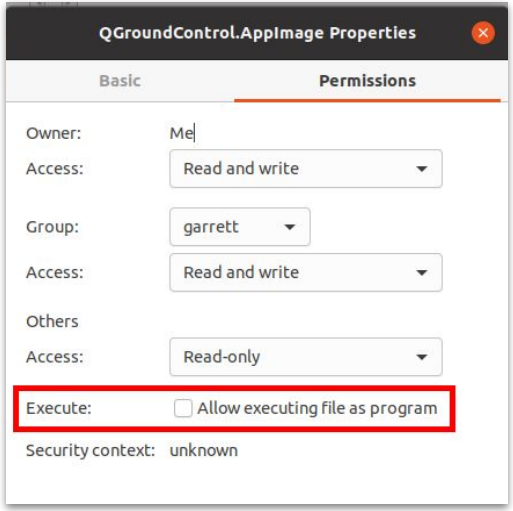
Installation: QGroundControl

Alternate permission changing for AppImage

1. Right click AppImage in file manager and select properties



2. In the permissions tab check the Execute Box.



Installation: ROS

Since we are using Ubuntu 20, ROS Noetic is used.

The following instructions are taken directly from the ROS wiki:
http://wiki.ros.org/noetic/Installation/Ubuntu

1. Set up your sources.list

   ```
   sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb_release -sc) main" > /etc/apt/sources.list.d/ros-latest.list'
   ```

2. Set up your keys

   ```
   sudo apt-key adv --keyserver 'hkp://keyserver.ubuntu.com:80' --recv-key C1CF6E31E6BADE8868B172B4F42ED6FBAB17C654
   ```

If the commands (1) and (2) fail, when copying directly from these instructions, it is likely due to the command not being copied correctly. The instructions need to be pasted as a single command (line)

Installation: ROS

3. Installation

   sudo apt update

   sudo apt install ros-noetic-desktop-full -y

4. Environment Setup

   echo "source /opt/ros/noetic/setup.bash" >> ~/.bashrc

   source ~/.bashrc

5. Dependencies for building packages

   sudo apt install python3-rosdep python3-rosinstall python3-rosinstall-generator python3-wstool python3-catkin-tools python3-osrf-pycommon build-essential -y

Installation: MAVROS

6. MAVROS

   sudo apt install ros-noetic-mavros -y

   wget https://raw.githubusercontent.com/mavlink/mavros/master/mavros/scripts/install_geographiclib_datasets.sh

   sudo bash ./install_geographiclib_datasets.sh

PX4/SITL

1. Python packages

   pip3 install toml numpy pyros-genmsg packaging jinja2

2. ROS workspace setup

   mkdir -p ~/leapfrog_ws/src && cd ~/leapfrog_ws

3. Clone LEAPFROG simulation repos

   # clone PX4 source code into ROS workspace src directory

   git clone https://github.com/SERC-LEAPFROG/LEAPFROG-Firmware --recursive src/LEAPFROG-PX4-Firmware

   # clone sitl_gazebo folder to be right under src directory

   git clone https://github.com/SERC-LEAPFROG/LEAPFROG-Simulation --recursive src/LEAPFROG-Simulation

   # clone mavlink headers with this exact path, so that they can be found by SITL

   git clone https://github.com/SERC-LEAPFROG/LEAPFROG-Mavlink src/mavlink/mavlink/v2.0

Usage

1. Build simulation
   a. Two main methods: catkin or catkin-tools
      i. catkin

```
# while in leapfrog_ws
catkin_make_isolated
source devel_isolated/setup.bash
```

      ii. catkin-tools

```
# while in catkin_ws
catkin config --extend /opt/ros/noetic  # only needed on first setup
catkin build
source devel/setup.bash
```

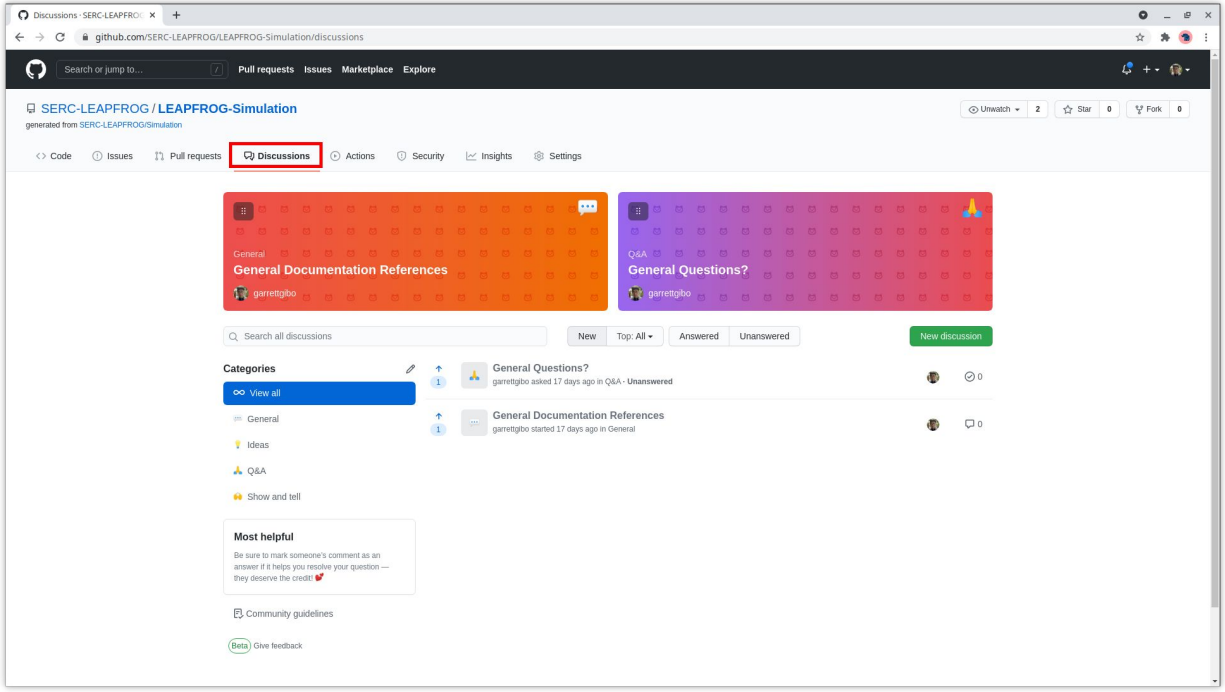
   b. A build from scratch can take >10 min (depending on the system that is being used), but subsequent builds should be much faster

Usage

2. Run simulation
   roslaunch px4 mavros_posix_sitl.launch
3. Configuration
   roslaunch px4 mavros_posix_sitl.launch gui:=true paused:=false

Name	Default	Description
gui	true	Starts up the simulation with the actual Gazebo GUI enabled. Setting this value to false, will run the simulation in headless mode, which can drastically increase runtime at the cost of no visual feedback.
paused	true	This value is only enabled when gui is set to true. When true, the Gazebo environment will start in a paused mode that prevents time from moving forward. This is helpful when inspecting the actual model of the vehicle without having to worry about the effects of the simulation affecting the state.

Get In Touch


Discussions · SERC-LEAPFROG
github.com/SERC-LEAPFROG/LEAPFROG-Simulation/discussions
Search or jump to…
Pull requests
Issues
Marketplace
Explore
SERC-LEAPFROG / LEAPFROG-Simulation
generated from SERC-LEAPFROG/Simulation
Unwatch 2
Star 0
Fork 0
Code
Issues
Pull requests
Discussions
Actions
Security
Insights
Settings
General
General Documentation References
garrettgibo
Q&A
General Questions?
garrettgibo
Search all discussions
New
Top: All
Answered
Unanswered
New discussion
Categories
View all
General
Ideas
Q&A
Show and tell
General Questions?
garrettgibo asked 17 days ago in Q&A · Unanswered
General Documentation References
garrettgibo started 17 days ago in General
Most helpful
Be sure to mark someone's comment as an answer if it helps you resolve your question — they deserve the credit!
Community guidelines
Beta
Give feedback

References

- Ubuntu 20
- Windows WSL
- VMware
- Gazebo
  - Installation
  - Github
- PlotJuggler
  - Snap
  - Github
- QGroundControl
- ROS Ubuntu Installation wiki

- LEAPFROG
  - Github Organization
  - Individual Repos
    - Firmware
    - Simulation
    - Mavlink
  - Setup Instructions