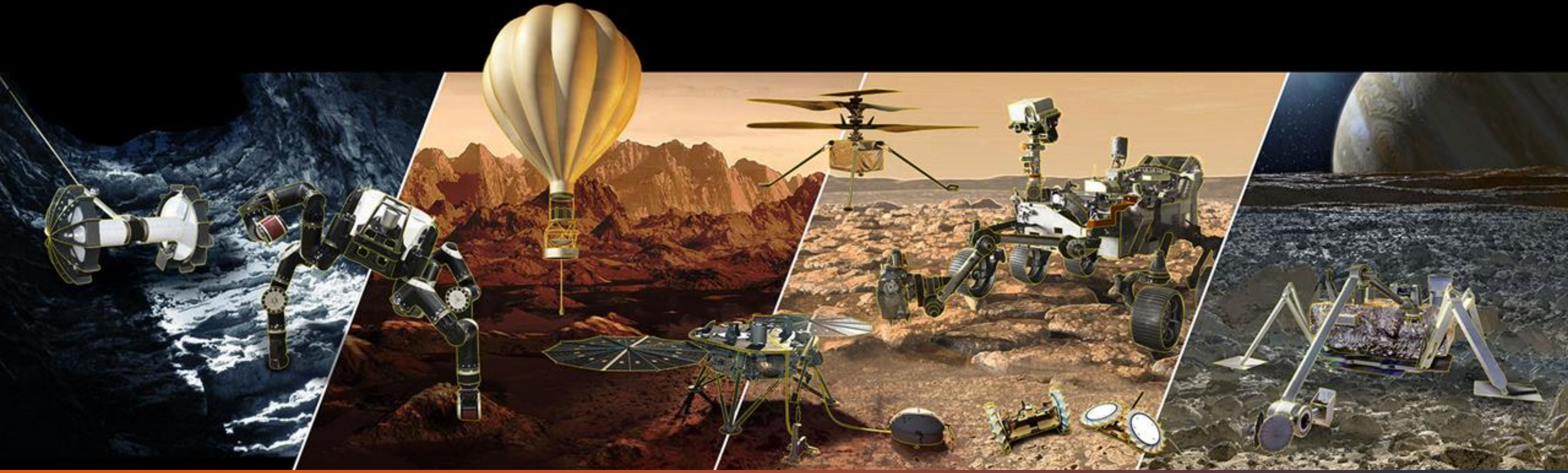
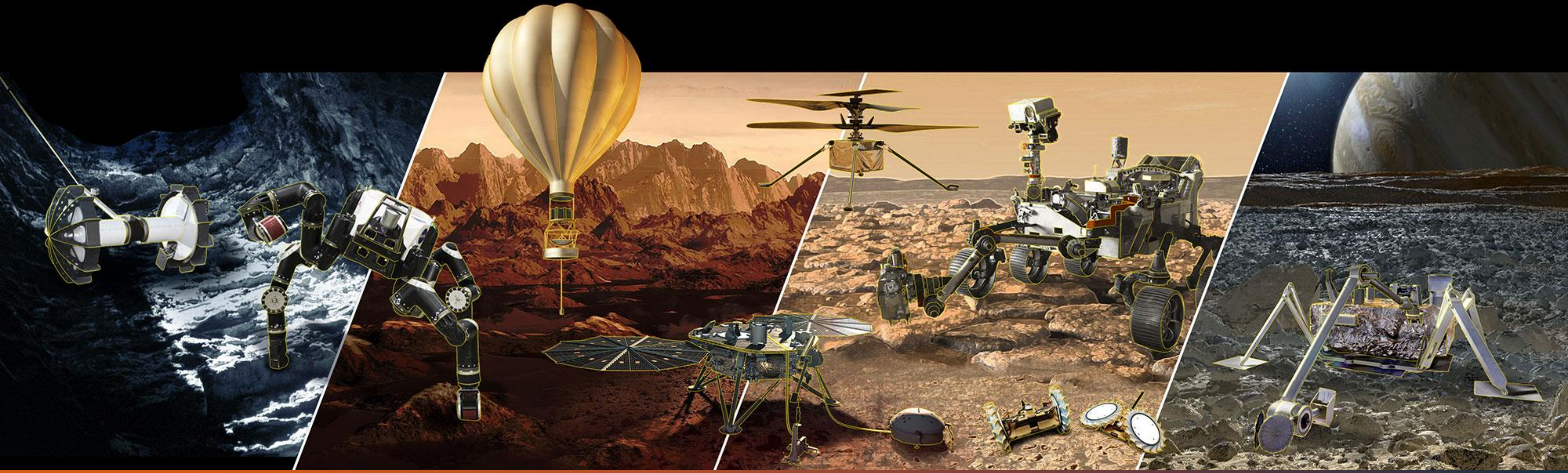


Mars 2020 Extended Operations

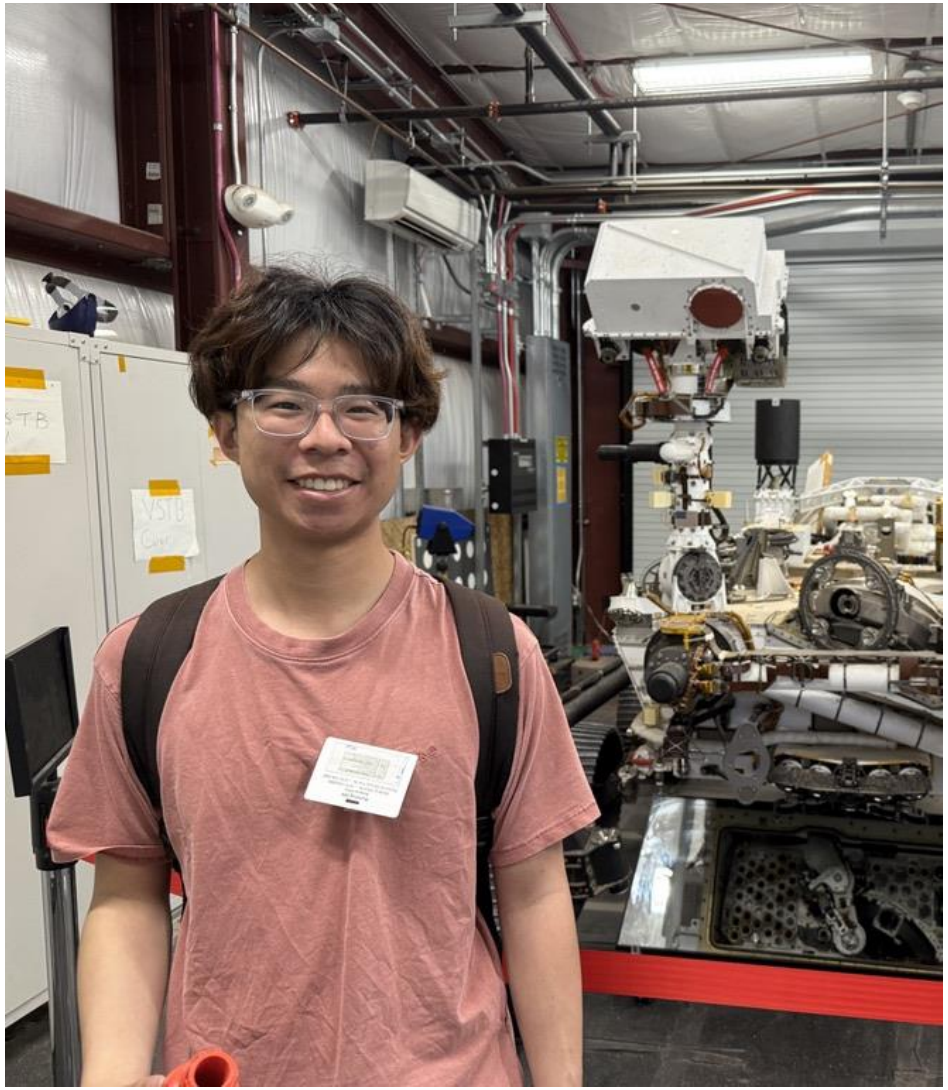
Automated Optimization of Safe Arm Abrasion Targets with AutoTarget

Matthew Jiang, 347R (Jet Propulsion Laboratory, California Institute of Technology)



Introduction

Matthew Jiang



- **Education:**
  - B.S. in Computer Engineering and Computer Science from University of Southern California
  - Robot Locomotion and Navigation Dynamics Laboratory (RoboLAND)
    - Undergraduate Researcher
  - USC Autonomous Underwater Vehicle Design Team
    - Software Lead
  - Incoming M.S. in Robotics at Georgia Institute of Technology
  - Incoming: Lunar Lab @ Georgia Tech (Lu's Navigation and Autonomous Robotics Lab)
    - Advisors: Prof. Lu Gan and Prof. Yongsheng Chen
- **Previous Experiences:**
  - Software Engineering Intern at Safran Passenger Innovations
    - In-flight entertainment and connectivity
  - Embedded Software Intern at Singular Medical USA
    - Implantable cardiac defibrillators and monitors

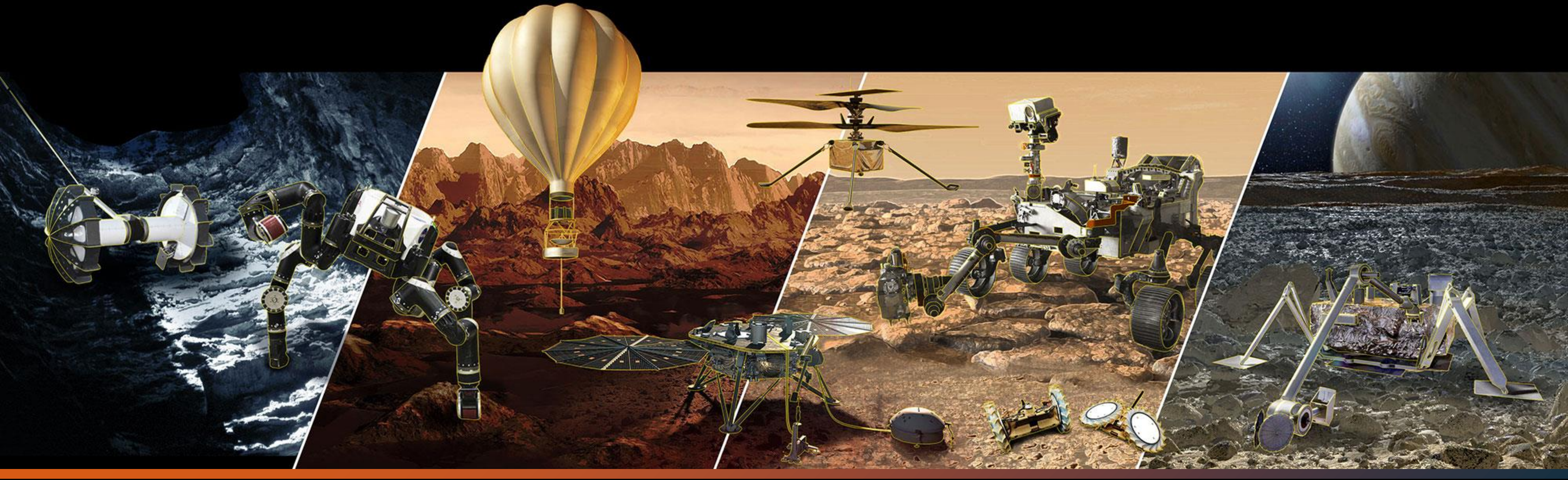
JPL Internship

From May 19 to August 1, 2025

Mentors:

Spencer Gregg – 347R

Justin Huang – 347K



Project Background

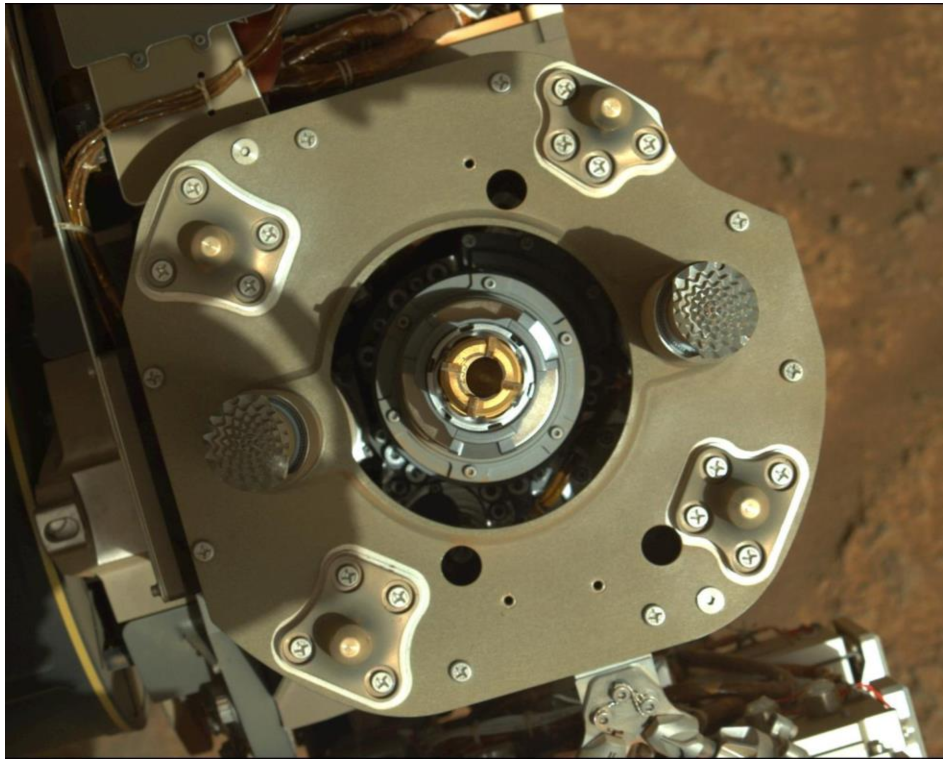
Mars 2020

- Mission Objectives:
  - Collect samples
  - Understand History of Mars
  - Detect biosignatures
- Perseverance Rover
  - 7 ft. robotic arm
  - Arm hardware:
    - Drill
      - To abrade and core
    - SHERLOC
      - ultraviolet Raman spectrometer
    - PIXL
      - X-ray fluorescence spectrometer
    - WATSON
      - Color camera
    - gDRT
      - Gas dust removal tool

Arm Abrasion

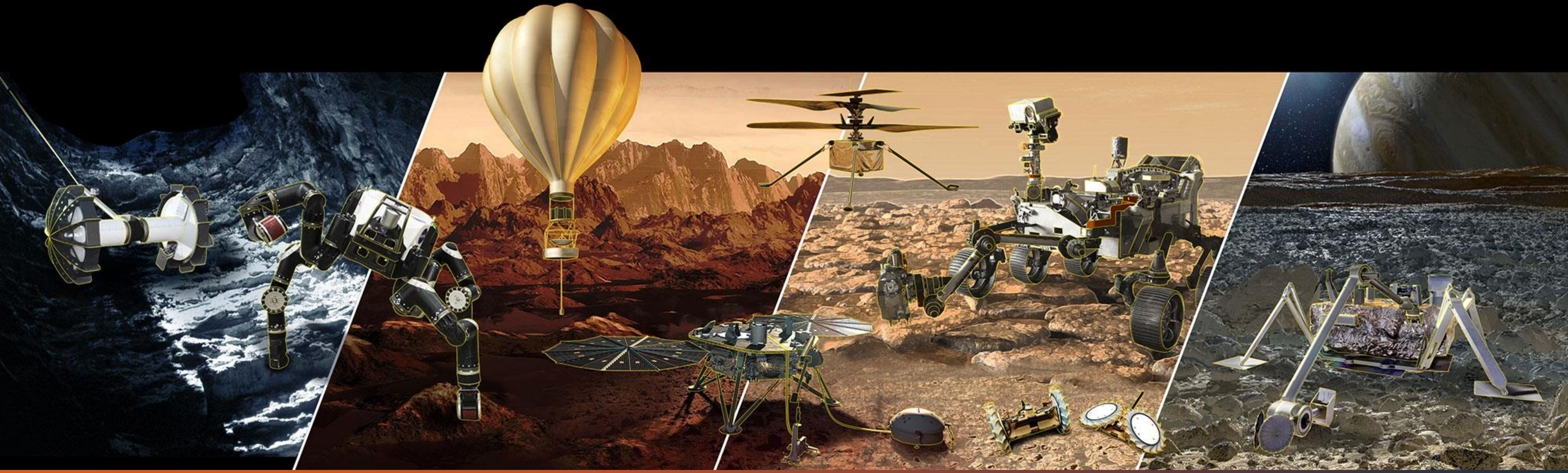
What is arm abrasion?	Why is arm abrasion difficult?
• Use a flat drill bit to remove weathered surface of rock for analysis and extraction • Scientists pick a rock and a rough target location • Rover Planners verify safety of target point and determine if the rover can abrade	• Scientific instruments are connected to the same arm as the abrading bit • Instruments have low margin for error • Commonly within 1 in. proximity to rock/terrain

Arm Activities



Use various instruments on an abraded surface to perform experiments, or drill and store rock core using coring bit

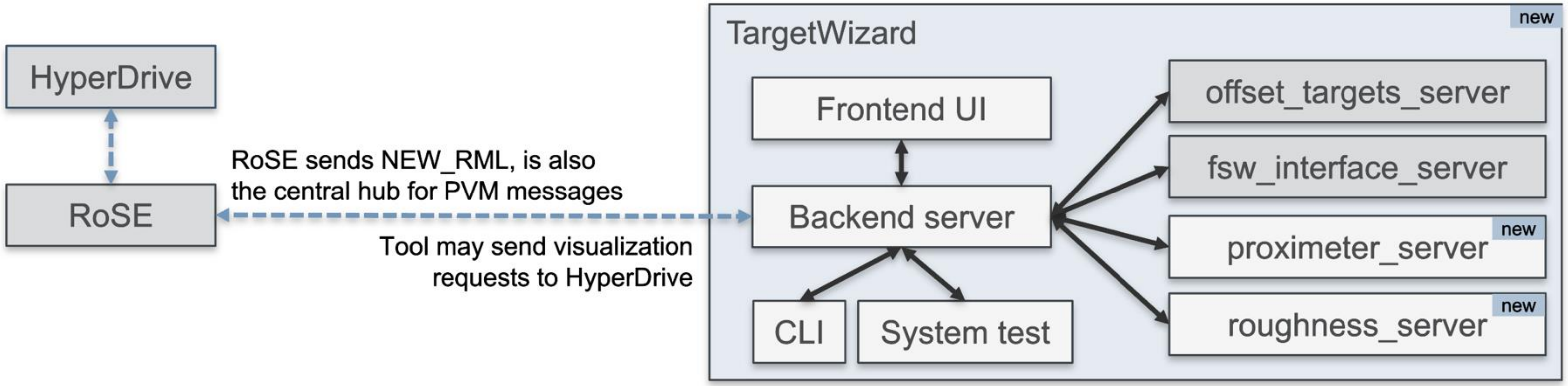
Arm activities target a precise location within an abrasion patch, a circle with a diameter of 50 mm



TargetWizard and AutoTarget

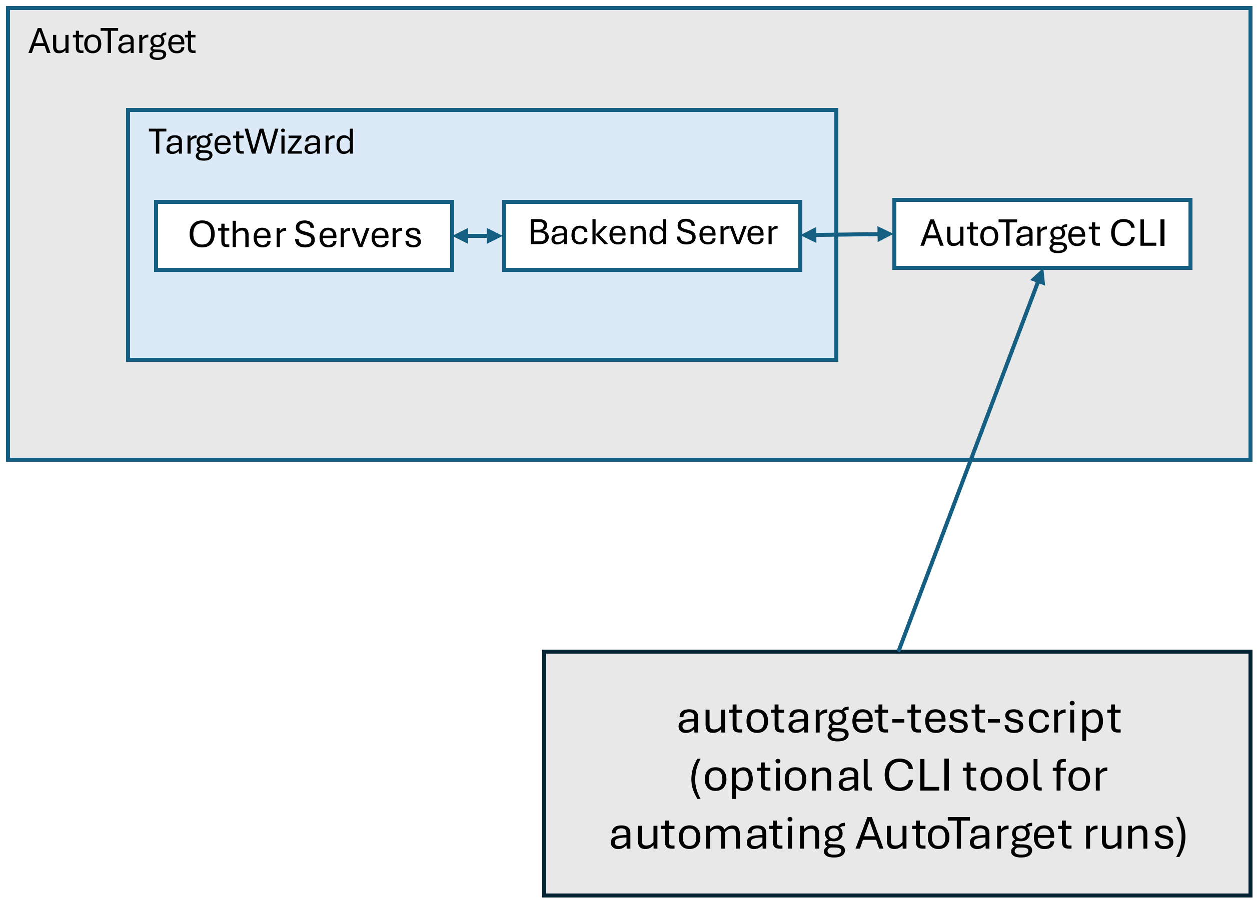
TargetWizard

- RSVP (Robot Sequencing and Visualization Programs) Tool
  - Automates the procedure for evaluating arm abrasion targets.
  - Consists of a GUI and backend services
  - Integrates and can be launched with RoSE
- Motivation:
  - Arm abrasion process is error prone and takes a long time per target (about 30 minutes)
  - TargetWizard shortens this process to under 10 seconds per target



TargetWizard Architecture Diagram. From Justin Huang, 2023

AutoTarget Architecture



AutoTarget

- AutoTarget utilizes TargetWizard’s backend server to evaluate potential targets for arm abrasions.
- After given a seed target, AutoTarget suggests potential replacement targets that may improve the **abrasion depth margin**
  - Abrasion depth margin: how deep the arm can abrade before contact with terrain

- AutoTarget pulls the terrain maps from the RML file and filters points in the point cloud
  - Primary workspace only
  - Radius
- It then randomizes the points and iterates through, scoring them based on abrasion depth margin margin

Early Improvements

Add option to use extended workspace

- Motivation: **Rover planners (RPs)** couldn’t run autotarget on points that were outside the primary workspace.
- Made the primary workspace filter optional

Add descriptive list of N targets

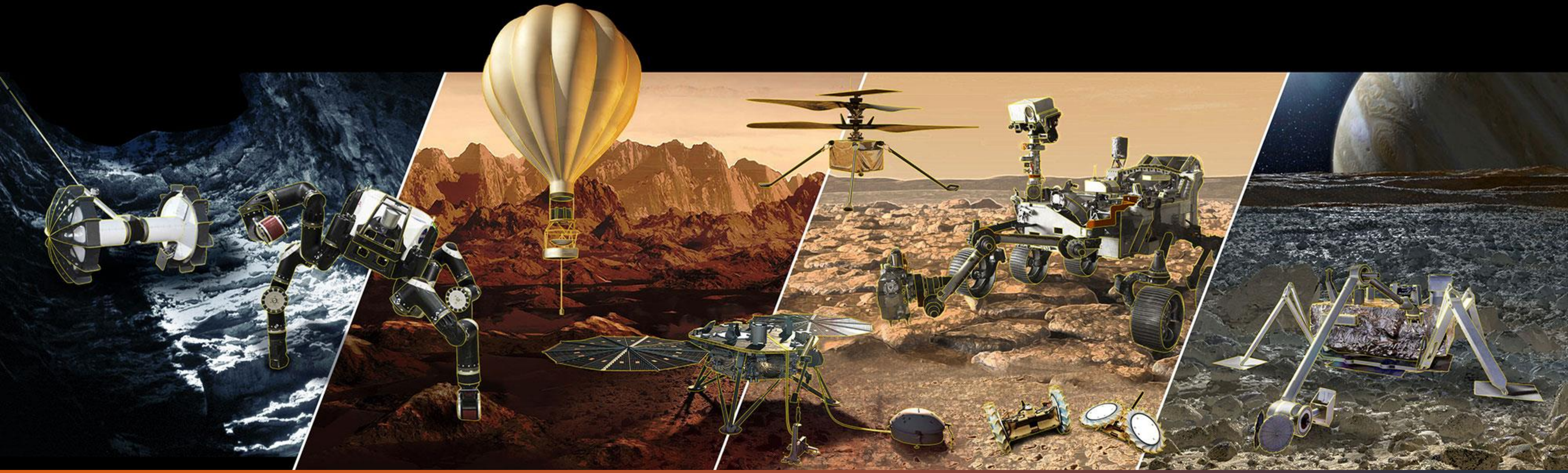
- Motivation: RP’s sometimes want to see if there are better targets that still improve the abrasion depth margin but are not necessarily the best margin
- Added a list of top N targets, N is configurable

Add stabilizer patch roughness filter

- Motivation: SNC requires targets to pass stabilizer path roughness threshold in order to perform drilling
- Query TargetWizard state to return stabilizer patch roughness
- Filter targets out of top N if check fails a required stabilizer roughness threshold in mm.

Implement Normal Tweaking

- Motivation: RPs sometimes have a good target but want to tweak the target normal vector only.
- Implementation to be described on the next slides



Normal Tweaking with AutoTarget

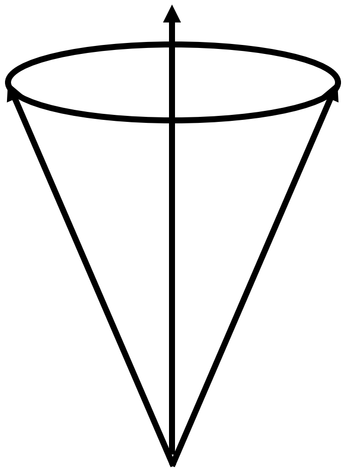
Normal Tweaking

RPs sometimes have a good target but want to tweak the target normal vector only.

Search a seed target location for alternative normal vectors

Given:

- Seed Target
- Maximum angle of deviation



Generate candidate points

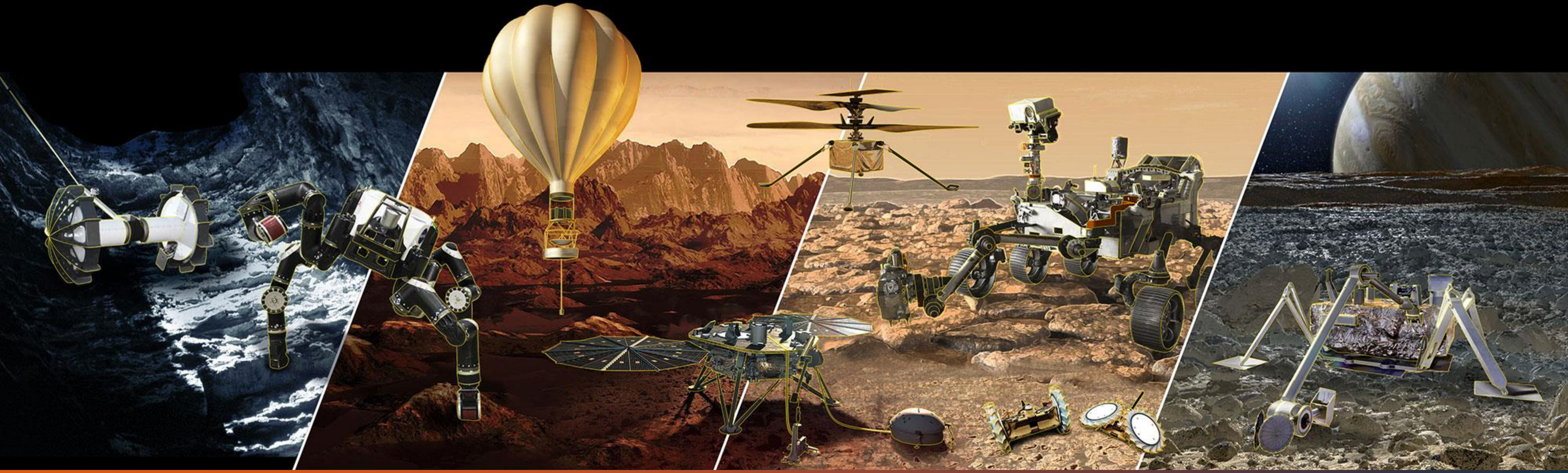
- Compute radius of candidate point circle
  $r = \tan(d\theta)$
- Generate points the circle orthogonal to the z-axis
  $p_r \sim \sqrt{\mathcal{U}(0, \tan(d\theta))},$
  $p_\theta \sim \mathcal{U}(0, 2\pi)$
- Compute a rotation matrix R to align with the target normal
- Rotate points using R
- Shift points down the axis by adding the original normal vector
- Normalize to unit vector

Normal Tweaking Usage

RPs use normal tweaking mode through the command line

- --mode normal
- --mode position (retains old behavior)
- --max-normal-tweak <angle-in-degrees>

Uniform random points are useful because we cover the highest possible resolution within the desired cone, and we don’t need to manually randomize them



Improving AutoTarget’s Efficiency

AutoTarget using Bayesian Optimization

- Target selection is an optimization problem
- Bayesian Optimization
  - Sequential optimization method that uses probabilistic models (gaussian processes) to efficiently find optimal values for **expensive-to-evaluate** functions by balancing exploration and exploitation.
  - Commonly used for hyperparameter tuning and black-box optimization where function evaluations are **costly or time-consuming.**

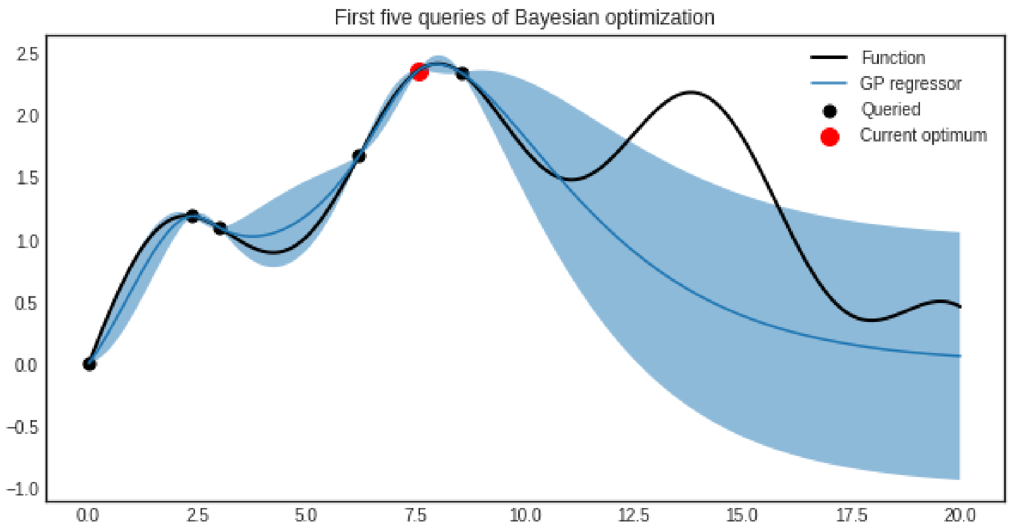


Image Source: https://modal-python.readthedocs.io/en/latest/content/examples/bayesian_optimization.html

TargetWizard queries are expensive

Each target takes ~6 to 7 seconds to evaluate

- 80 targets in 10 minutes

RPs run AutoTarget for ~10 minutes, want to reduce to less

We should be intentional with what targets we choose to evaluate!

TargetWizard uses C++11

RSVP development machines are old

- TargetWizard uses C++11

No good existing libraries that implement gaussian process in C++11

Write our own library, called **gaussian-process**

gaussian-process

- Our gaussian process library implements a Radial Basis Function (RBF) kernel, one of the most used kernels
  - Requires assumption that terrain follows gaussian distribution
- Provides functions:
  - Add data point, returns void
  - Predict a vector of inputs, returns GP posterior
- Supports n dimensions
- Supports C++11

$$k(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right)$$

Bayesian Optimization Implementation

Mode specific parameters:

- Variance
- Lengthscale of the kernel
- Note: since normal vectors are unit vectors, lengthscale and variance parameters effectively capture the angle in radians

Non-mode specific parameters

- β: scales the confidence bounds to match a z-score of confidence
  - i.e. β = 2 means 97% confidence that actual value is within the confidence bounds

Optimization Method

- Largely based on GP-UCB.
- Confidence intervals $c(i) = \mu(i) \pm \sqrt{\beta}\sigma$
- Potential Maxima are points where the upper bound is higher than the highest lower bound.
- Pick the candidate with highest confidence width.
- If there no potential maximizers, pick the highest confidence width.

Local maxima

Random sampling is more versatile than Bayesian Optimization

- Slow, guaranteed convergence to the actual maximum
- Bayesian Optimization can quickly find local maxima and get stuck

Combine the two methods using Epsilon-Greedy

- Compromise efficiency for versatility

Result:

- Sometimes smarter" exploration and exploitation
- Sometimes more random with more freedom
- A new tunable "temperature" parameter τ

AutoTarget Test Script

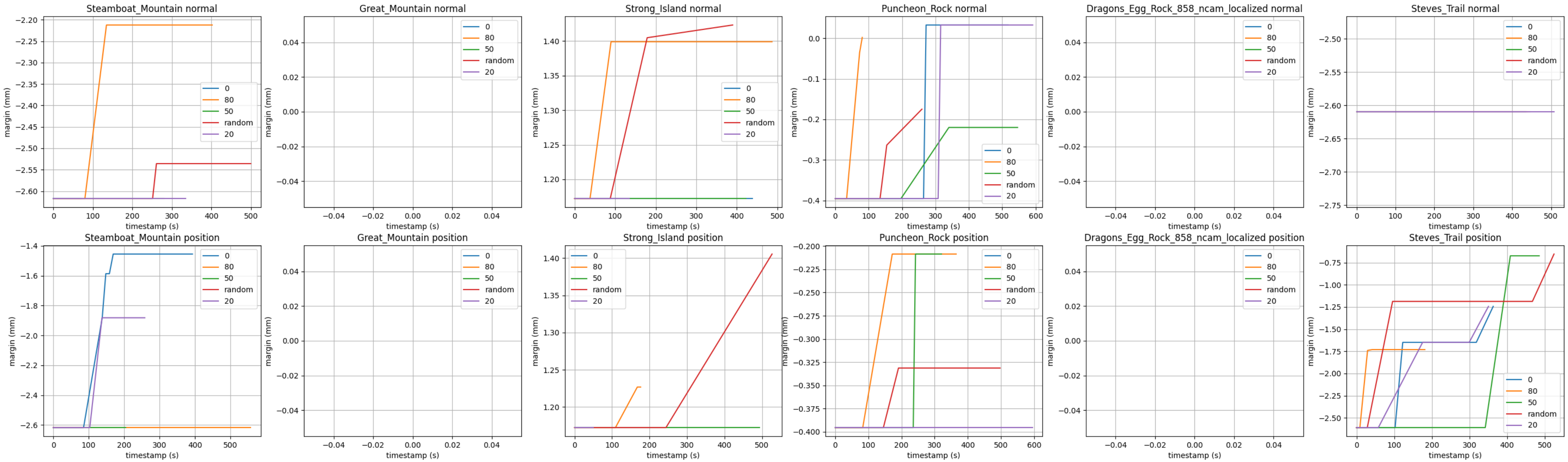
Bash script that runs multiple instances of AutoTarget in parallel

6 instances of AutoTarget: negligible slowdown

Input: parameter list file

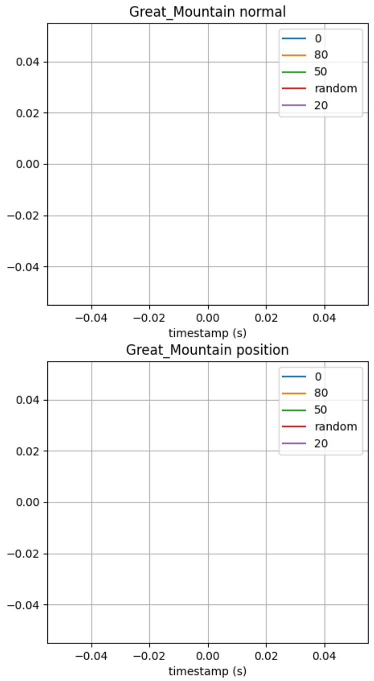
Bayesian Optimization in Practice

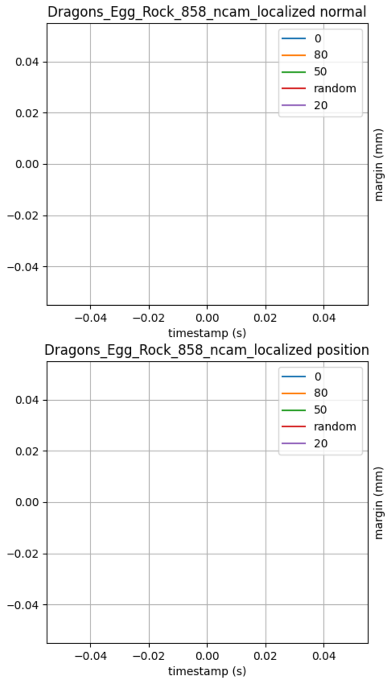
- Experiments were conducted on 6 real past targets wit h a maximum experiment length of 10 minutes with both position tweak and normal tweak, on temperature values {0, 0.2, 0.5, 0.8, 1}
  - 3 difficult (Great Mountain, Steamboat Mountain, Dragons Egg Rock)
  - 3 easy (Puncheon Rock, Strong Island, Steves Trail)



Bayesian Optimization in Practice

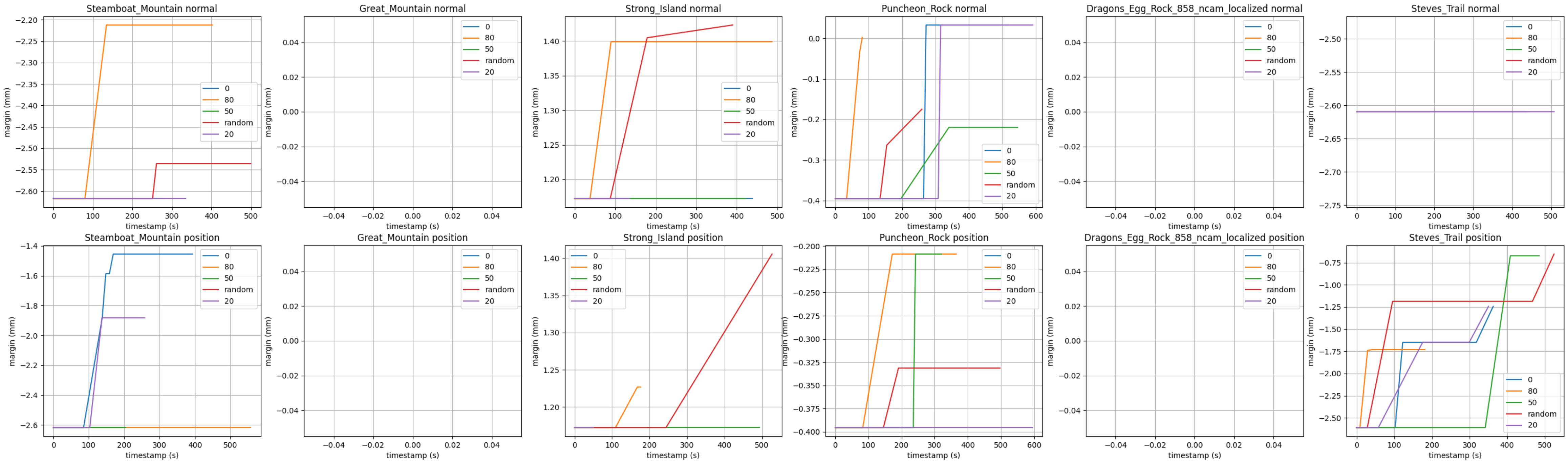
- Two targets, Great Mountain and Dragons Egg Rock were very challenging targets that the baseline (Random Sampling) could not improve over a 10 min period
  - Bayesian Optimization also could not find an improvement





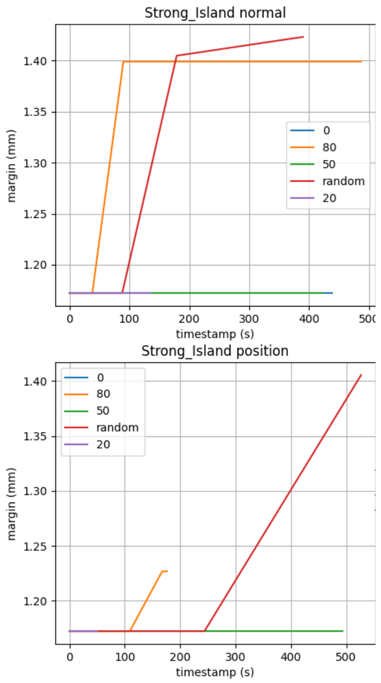
Bayesian Optimization in Practice

- Considering an experiment length of 2 minutes, random sampling (red) is beat by some temperature variant of Bayesian optimization



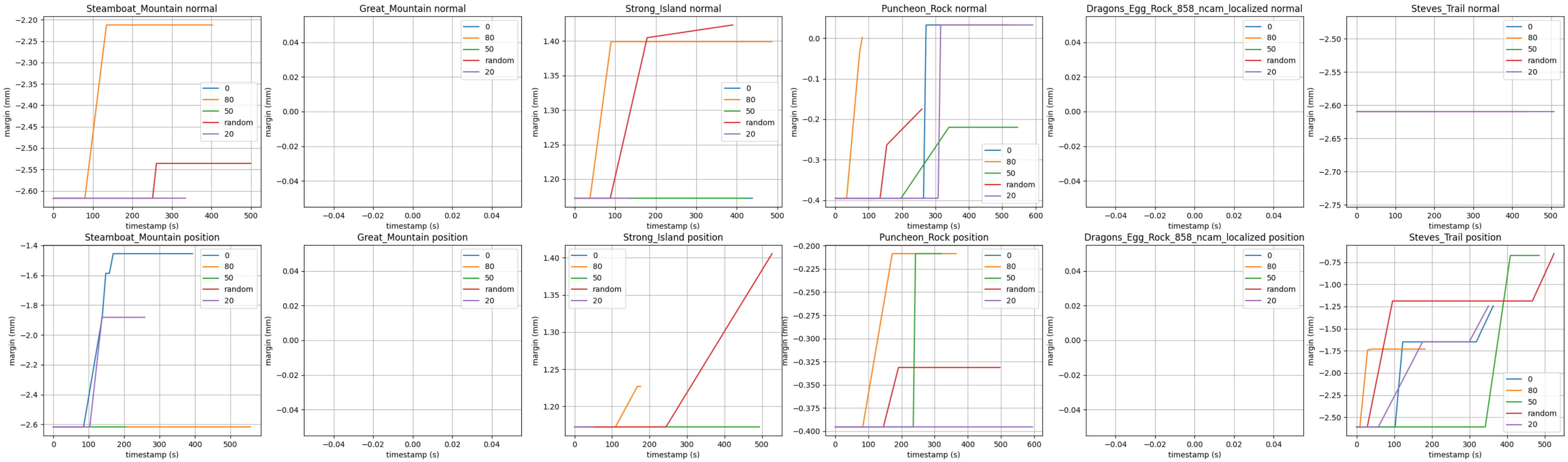
Bayesian Optimization in Practice

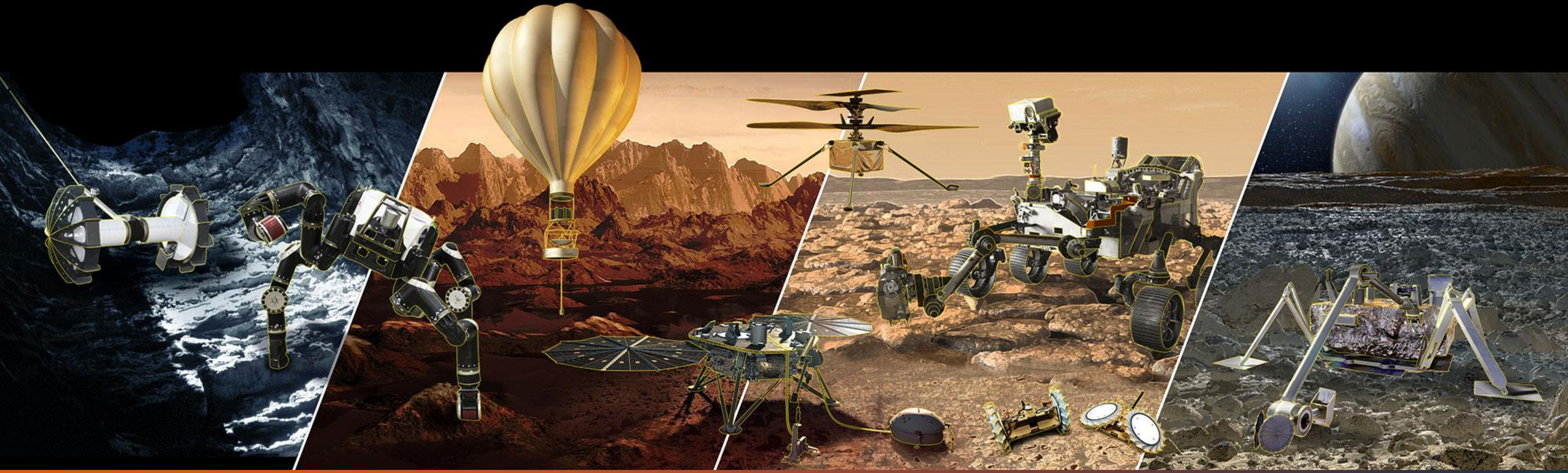
- Past 2 minutes, random sampling can sometimes beat Bayesian Optimization



Bayesian Optimization in Practice

- Conclusion: it is difficult to compare the two methods with such little data since random sampling is very **random**
- Can we conclude that Bayesian optimization necessarily beats random sampling?
  - No, we can't since we don't have enough data





AutoTarget for Arm Activities

Arm Activities

Implementation of AutoTarget for Arm Activities

- Moved on due to time constraints
- RPs also want to use AutoTarget to help optimize targets for Arm Activities

Arm Activities has a lot of required parameters

- Too many parameters makes using a CLI tool very cumbersome and unintuitive

Arm Activities

List of Arm Activities

- Drill
- SHERLOC
- PIXL
- WATSON
- gDRT

Each one has different parameters to configure before evaluation

Revamp AutoTarget to reduce input parameters

Ideal: Configure AutoTarget within TargetWizard's interface

- Integrate AutoTarget into TargetWizard directly (time consuming project)
- Allow state transfer from TargetWizard to AutoTarget through save files (faster deployment)

Dump TargetWizard state using a GUI button

- Allows saving the state to a file somewhere on the computer

Configure AutoTarget using state file

- --state <file-path>
- This new argument replaces all old arguments except:
  - --rml <rml-file> (to load terrains)
  - Other AutoTarget specific parameters

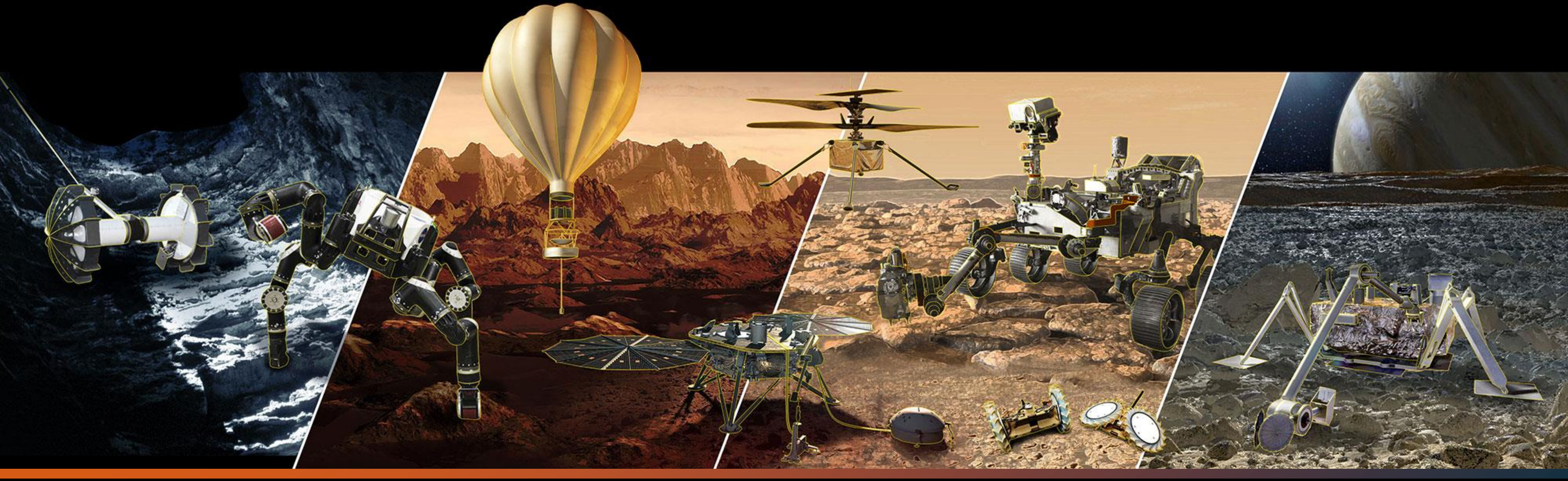
Arm Activities Modes

Normal Tweak

- Same implementation as Arm Abrasion except for loading different evaluation configuration

Position Tweak

- More complex as required positional accuracy is higher
- Use periscope to localize rover camera terrains
- Left unimplemented due to time constraints



Legacy

AutoTarget Usage in Rover Operations

Sol 1504:
- AutoTarget found better target
- Failed SNC assessment due to stabilizer roughness

Sol 1513:
- 3.3mm → 4.6mm target found by AutoTarget
- Manual search needed for extended workspace

Sol 1525:
- 1.4mm target found using TargetWizard
- AutoTarget found 1.6mm margin option but it was not used

Sol 1544:
- AutoTarget maintained 2.0mm margin
- Normal adjustment added 0.1-0.2mm

Sol 1565:
- No margin → -4.9mm margin
- Required 2,000 AutoTarget assessments on three different seed targets

Future Projects

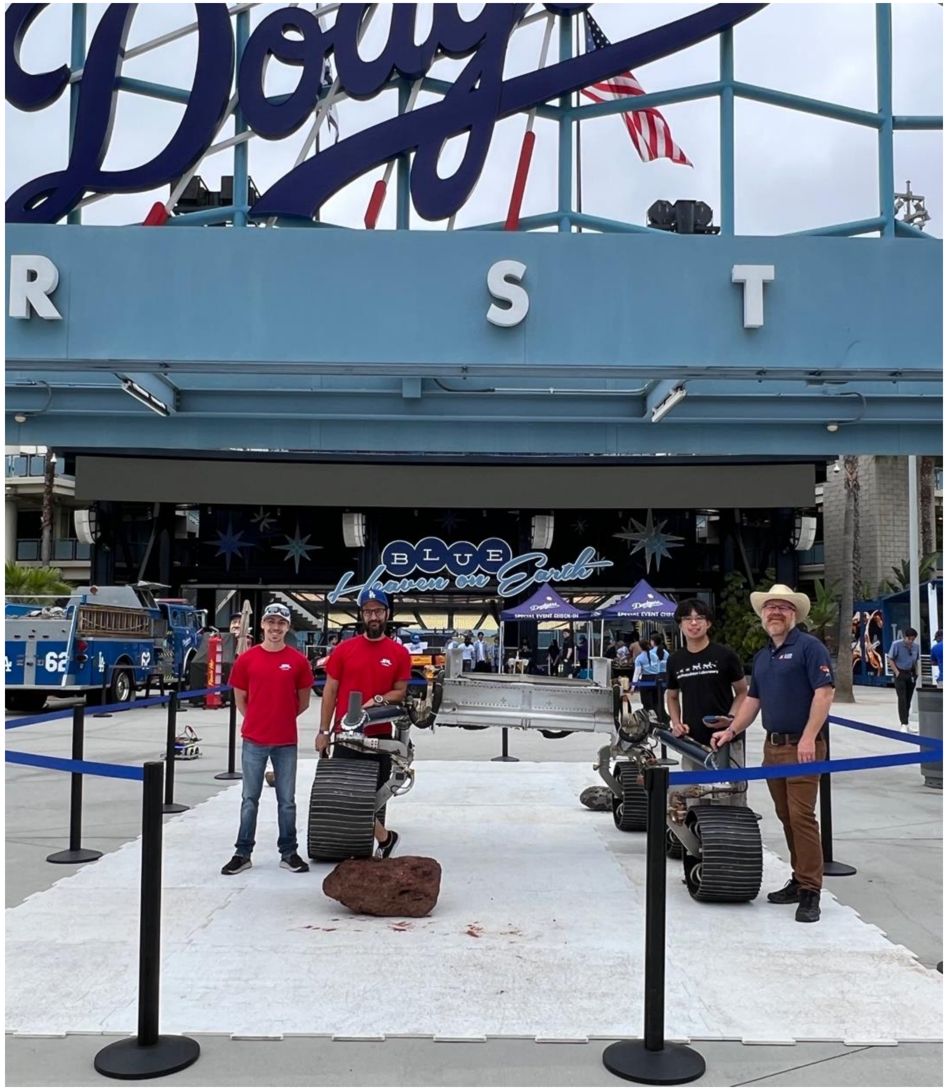
Continue experiments on Bayesian Optimization with a higher sample size

Complete position tweaking implementation for Arm Activities

Integrate AutoTarget into TargetWizard GUI

Explore optimization strategies that don't require evaluations for normal tweaks using knowledge of robot geometry

Other Ventures



Outreach at Dodger Stadium for JPL STEM day

- Invigorate curious adults and children
- Get asked many times if aliens exist
- Acquire a coveted JPL Dodgers bucket hat

Acknowledgements

Spencer Gregg

Justin Huang

Philip Twu

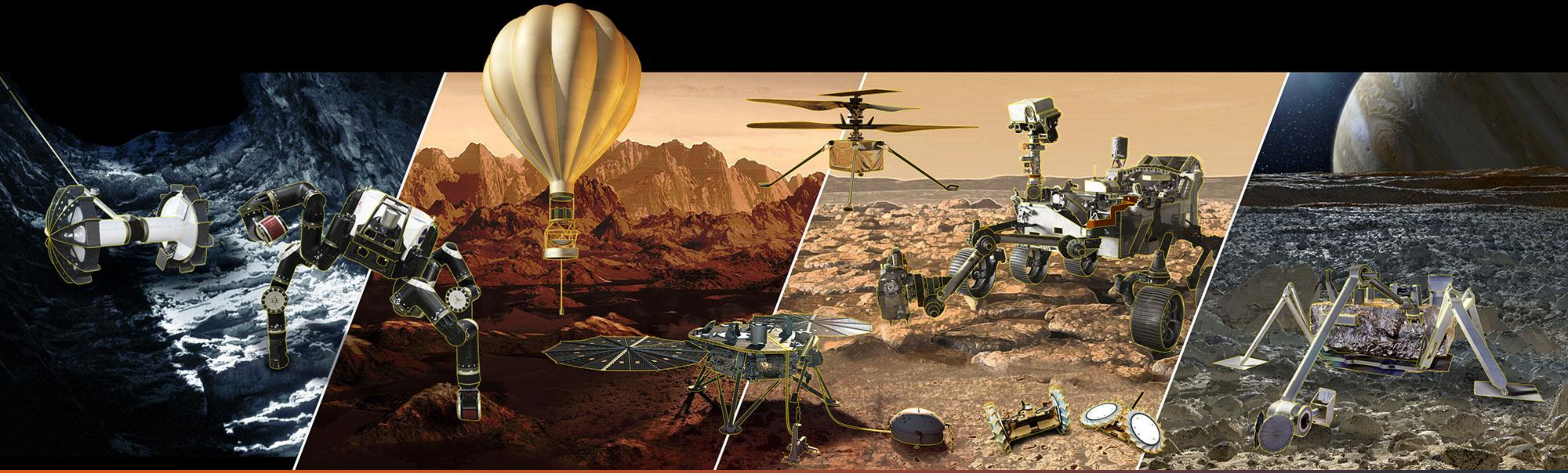
Bret Witt (For helping to brainstorm)

RSVP Team

And many more…

What's next for me

- M.S. Robotics at Georgia Institute of Technology
  - Researching applications of robotic perception/navigation in agriculture



Thank You!

JPL