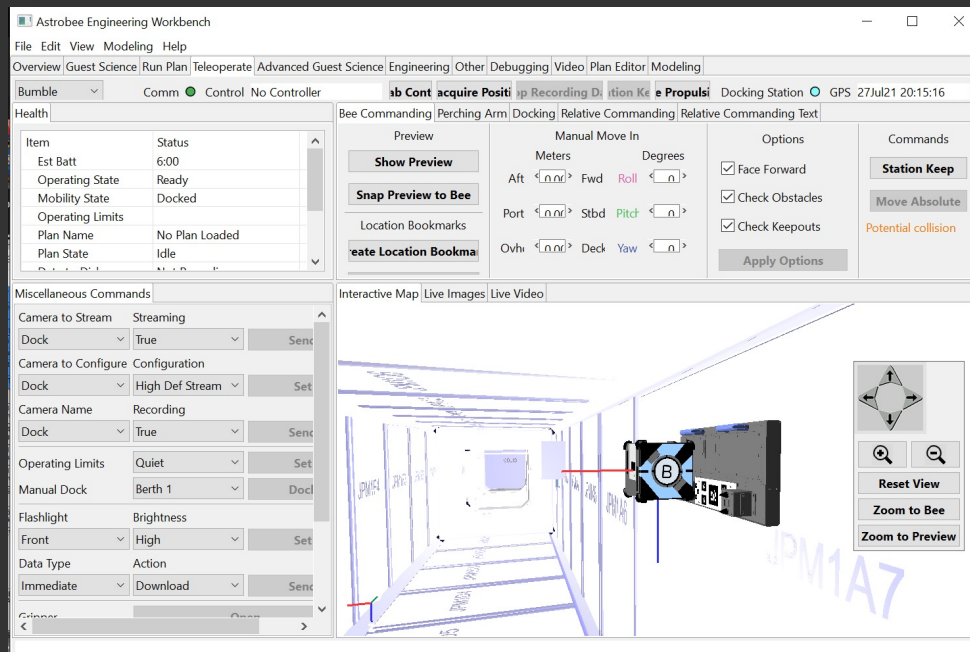


Revamping Astrobbee's Ground Data System

Moez Khan | Michigan State University

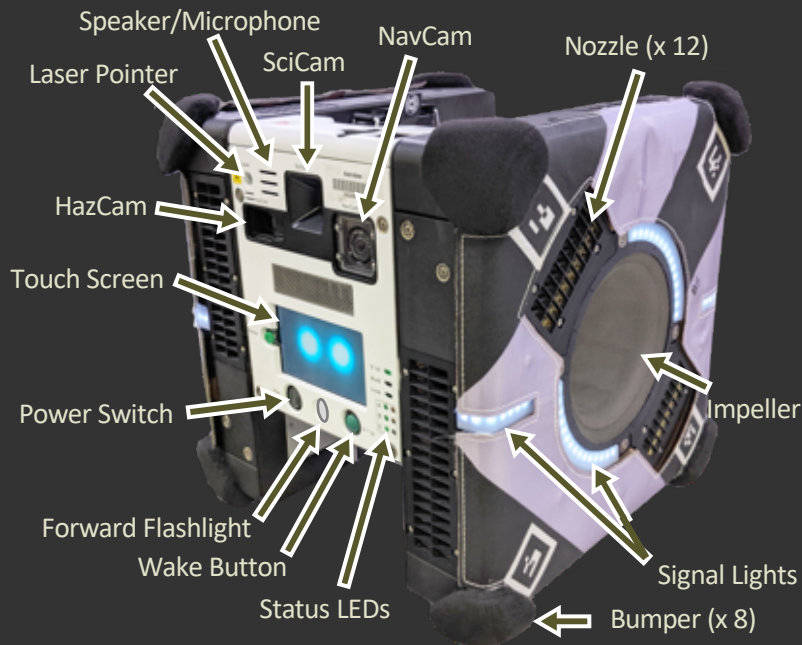
Mentor: Jose Benavides, NASA Ames Research Center - TI

Session: Summer 2021



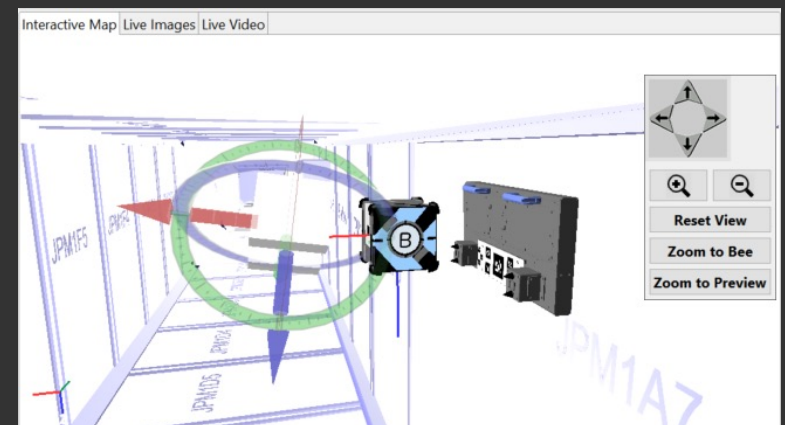
What is Astrobees?

- Astrobees is a system of 3 free-flying robots in the ISS designed to enable a wide variety of potential guest science and help astronauts with their daily tasks.
 - Keeping track of inventory
 - Maintain spacecraft
 - Moving cargo
 - Documenting experiments
- Bosch (Soundsee)
 - Can use audio and AI to monitor equipment/machine health.
- NPS (Astrobatics)
 - Utilizes a robotic arm to 'hop' from location to location
- Stanford (GECKO)
 - Uses gecko-inspired adhesion that can be used to perch robot on a wall



What is GDS?

- GDS or Ground Data System is Astrobees' control station software. Using GDS, an operator can command and monitor an Astrobee robot from the ground to space. GDS is a complete software but requires several bug fixes and improvements. My key objective during this internship was to significantly improve the software by implementing robust bug fixes and adding powerful features to enrich the experience of using GDS
 - Developed as an Eclipse RCP application (Java)
 - Has a complete GUI
 - Can command and monitor up to three robots
 - Fundamentally based on the Visual Environment for Remote Virtual Exploration (VERVE)



GDS Structure

- GDS has 11 different tabs used for various purposes. Some of the major tabs are:
 - Overview:** Provides a summary of the status of available robots and a 3D view. Also can hibernate and wake the robot.

The screenshot displays the GDS Overview tab. At the top, a navigation bar includes tabs for Overview, Guest Science, Run Plan, Teleoperate, Advanced Guest Science, Engineering, Other, Debugging, Video, Plan Editor, and Modeling. The Overview tab is active. Below the navigation bar, the status bar shows 'Docking Station' with a green indicator, 'GPS', and the timestamp '28Jul21 16:52:28'. The main content area is divided into two sections. The left section, titled 'Bee Status', contains a table with columns for Name, Comm, Health, Est Batt, and Control. The table lists three robots: Bumble, Honey, and Queen. Bumble has a green Comm and Health indicator, an estimated battery of 5:50, and is labeled 'No Controller'. Honey and Queen have blue Comm and Health indicators. Below the table is a 'Health Details' button. The right section, titled 'Live Map', shows a 3D perspective view of the docking station structure. The structure is a long, narrow, light blue rectangular box with a dark blue interior. A small robot icon is visible on the floor. The floor is labeled with 'JPM1A2' through 'JPM1A7' and 'Pit'. A coordinate system with X, Y, and Z axes is shown in the bottom left corner. On the right side of the Live Map, there are navigation controls: a directional pad, zoom in (+) and zoom out (-) buttons, and a 'Reset View' button.

Name	Comm	Health	Est Batt	Control
Bumble	●	●	5:50	No Controller
Honey	●	●		
Queen	●	●		

Docking Station Status and Commanding

Left Berth
Occupant: [Red Bar] Status: [Red Bar]

Right Berth
Occupant: [Red Bar] Status: [Red Bar]

Wake Commanding
[Dropdown] [Wake]

Hibernate Commanding
Bumble [Dropdown] [Grab Control] [Hibernate]

GDS Structure

- **Guest Science:** Provides some teleoperation commands and loading guest science apps

Overview
Guest Science
Run Plan
Teleoperate
Advanced Guest Science
Engineering
Other
Debugging
Video
Plan Editor
Modeling

Docking Station ● GPS 28Jul21 16:55:00

Bee Selection and Status

Name	Comn	Health	Est Ba	Control	Summary	Plan	Plan Status
☐ Bumble	●	●	5:40	No Controller		No Plan Loaded	Idle
☐ Honey	●	●					
☐ Queen	●	●					

Health Details

Commanding for -

Grab Control

Plans

Select and Load Plans...

Run Plans

Station Keep Bees

Payload Commanding

Guest Science Application Command

Send Command

Live Map
Live Images
Live Video

↑

↓

←

→

+

-

Reset View

Zoom to Bee

01:40:40 Bumble: Undock Completed

GDS Structure

- **Run Plan:** Can load and run pre-made plans that can configure various options for the robots and move the robot along a path.

Overview Guest Science Run Plan Teleoperate Advanced Guest Science Engineering Other Debugging Video Plan Editor Modeling

Bumble Comm ● Control makhan1@ARLWL202105 ● Require Pos: ● Recording on ● Docks Docking Station ● GPS 28Jul21 17:02:30

Health

Item	Status
Est Batt	5:00
Operating State	Ready
Mobility State	Stopped
Operating Limits	iss_nominal

Bee Commanding

Grab Control

Select Plan... C:\Users\makhan1\Documents\startup.fplan

Plan Valid

Load Run Pause Skip Step

Description

Live Plan

Plan Name startup

Plan State Idle

Total Elapsed Time 00:00:00

Plan Step	Duration
▼ startup	
▼ 0 Station	
0.0 SetCamera	
0.1 SetCamera	
0.2 SetCamera	
0.3 SetCamera	
0.4 SetCameraRecc	
0.5 SetCameraRecc	
0.6 SetCameraStre	
0.7 SetCameraStre	

Live Map Live Images Live Video

13:44:32 Bumble: Run Plan Completed

GDS Structure

- **Teleoperate:** Main tab for sending teleoperate commands to the robot and configuring some options

The screenshot displays the GDS (Guest Science Desktop) interface, specifically the **Teleoperate** tab. The interface is divided into several sections:

- Top Menu Bar:** Overview, Guest Science, Run Plan, Teleoperate, Advanced Guest Science, Engineering, Other, Debugging, Video, Plan Editor, Modeling.
- Status Bar:** Bumble, Comm (green dot), Control, makhan1@ARLWL202105, Coquire Po, Recording on, Docking Station (blue dot), GPS 28Jul21 17:04:38.
- Health Panel:**

Item	Status
Est Batt	4:50
Operating State	Teleoperation
Mobility State	Flying
Operating Limits	iss_nominal
Plan Name	startup
- Control Panels:**
 - Bee Commanding:** Preview (Hide Preview, Preview to), Location Bookm (Location Boc).
 - Manual Mov:** Meters, Degree: Aft < 11.0, Fw < 11.0, Por < 9.8, Stb < 5.8, Ovl < 1.5, Dev < 1.36, Yav < 1.36.
 - Options:** Face Forward (checked), Check Obstacles (checked), Check Keepouts (checked), Apply Options.
 - Commands:** Station Keep, Move Absolute.
- Miscellaneous Commands:**
 - Operating Limits: Nominal
 - Manual Dock: Berth 1
 - Flashlight: Brightness (Front, High)
 - Data Type: Action (Immediate, Download)
 - Gripper: Open
 - Idle Propulsion: Idle
 - Stop Arm: Stop Ar
 - Undock: Undoc
- Interactive Map:** Live Images, Live Video. The map shows a 3D perspective view of the robot (Bumble) in a simulated environment. The robot is a small black and white cube with a blue 'B' on top. It is positioned in the center of a green circular area. The map includes a compass rose, zoom in/out buttons, and buttons for Reset View, Zoom to Bee, and Zoom to Preview.
- Status Bar:** 16:59:20 Bumble: I S S Translate: 11.07, -9.81, 4.52; Rotate: 0.01, 0.77, 0.32, 0.56 Pending ...

Bug/Feature #1

- **Description:** after long LOS, some of the state-machines out of sync (states show as blank on GDS)
- **Steps to reproduce:**
 1. Launch GDS and connect to FSW
 2. Invoke a LOS
 3. Re-establish connection
- **Solution:** After returning from a LOS, the message that tells GDS to update its mobility states never gets sent anymore until you do something like input a teleop command. However, it does send messages for the position sample. I added a check before updating the position sample to see if the mobility states also need to be updated.

```
private AgentState prevAgentState = null;
```

```
if (agentState != null)  
    prevAgentState = agentState; // save the states in case we need to revert
```

Bumble		Comm ● Control	ab Contr	acc
Item	Status			
Est Batt	5:30			
Control				
Operating State				
Plan Execution State				
Raw Mobility State				
Sub Mobility State				
Operating Limits				
Fault ID	Description	Subsyst...	Node	

Bumble Data to GD

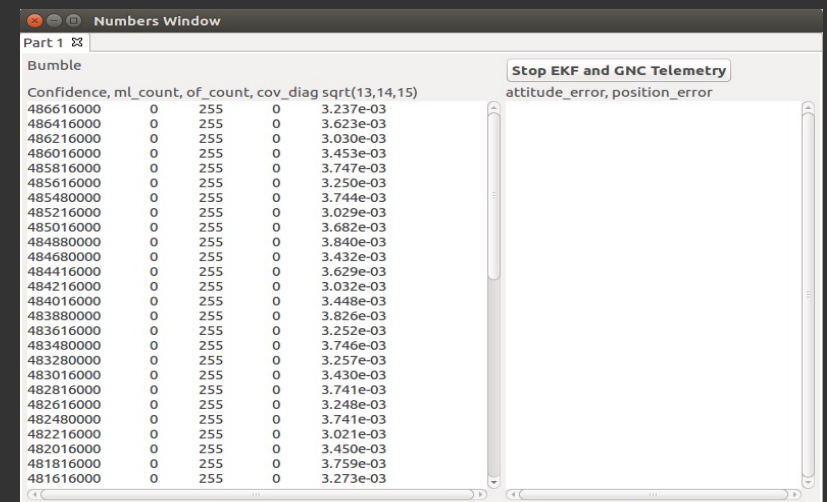
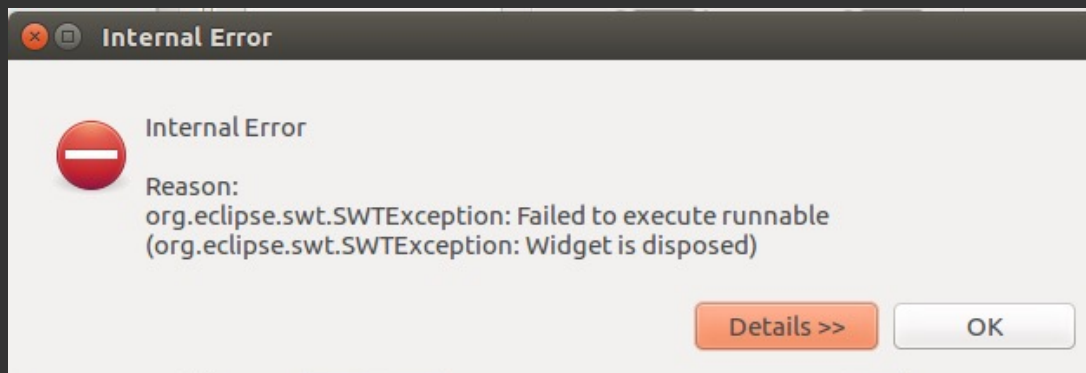
Telemetry	Current
Position	30.0 H
EkfState	0.0 Hz
CommStatus	1.0 Hz
DiskState	1.0 Hz
CpuState	0.0 Hz

Bumble Data to Disk

```
else if(msgType.equals(MessageType.POSITION_SAMPLE_TYPE)) {  
    // After returning from LOS, a bug causes the agentState message to not send  
    // But, the position sample message will send. So here, we check if the mobility state needs  
    // to be updated and if so, we update it with the last known valid state  
    if (aggregateState.getAstrobeeState().getMobilityState() == null) {  
        if (prevAgentState != null)  
            aggregateState.ingestAgentState(prevAgentState);  
        notifyListeners(MessageTypeExtAstro.AGENT_STATE_TYPE);  
    }  
}
```

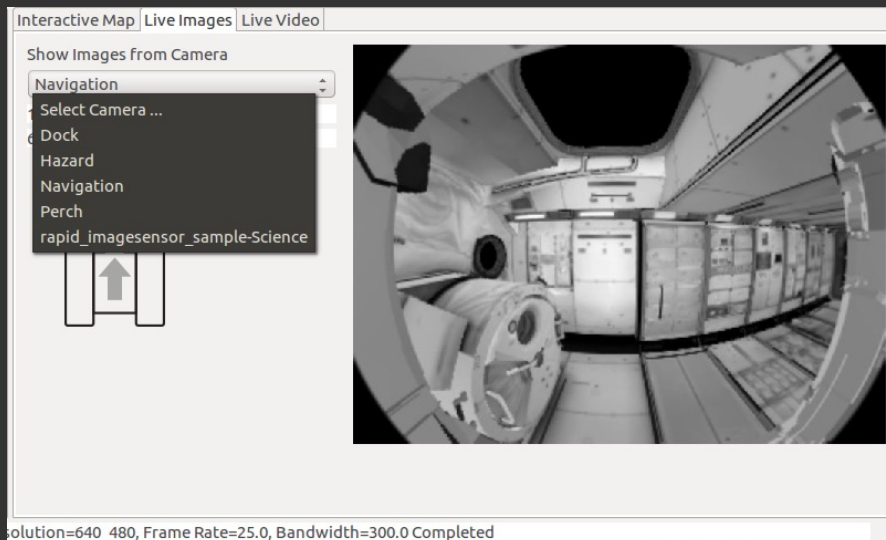
Bug/Feature #2

- **Description:** In certain conditions, the Numbers Window menu will cause an infinite SWT error box that leaves GDS in an inoperable state. Can only be fixed by force closing GDS.
- **Steps to reproduce:**
 1. Launch FSW and GDS and select Grab Control
 2. Run startup.fplan (didn't test any other plans, not sure if its specific to this plan or if its because of an option it enables)
 3. On menu bar: View → Numbers Window
 4. Close window
- **Solution:** The reason this bug happens is GDS doesn't properly check if a particular widget is disposed before doing an operation to it. This operation is done many times a second which causes the error to show up rapidly and infinitely. My solution was to simply correct the corresponding If statements to check if the widget is Disposed.



Bug/Feature #3

- **Description:** If you are on the Live Images subtab and then run the FSW, the dropdown menu will not be populated. This can only then be fixed if you restart either GDS or FSW.
- **Steps to reproduce:**
 1. Launch GDS and navigate to Live Images subtab
 2. Start FSW
- **Solution:** My solution was to create a function that is ran every time the Live Images subtab is made visible. This function will attempt to re-populate the dropdown menu. Therefore, you only need to revisit the tab for the dropdown menu to correctly repopulate.



```
public void partVisible(MPart arg0) {  
    if(agent != null) {  
        String prev = topicsCombo.getText(); // save the current camera we are on  
        topicsComboPopulated = populateTopicsCombo(); // re-populate camera list  
        topicsCombo.setText(prev); // set the camera we were previously on  
    }  
}
```

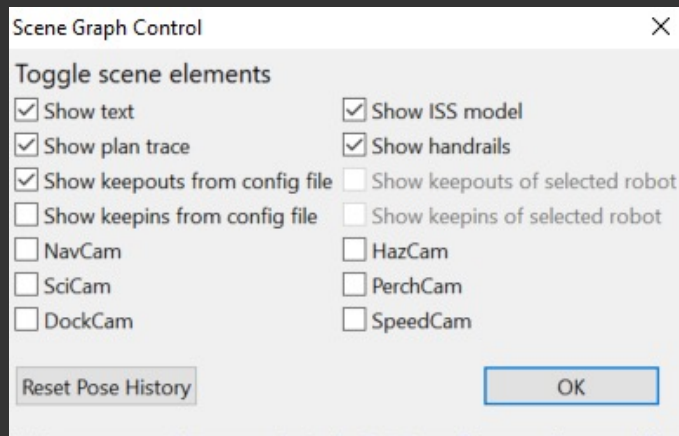
```
protected EPartService eps;
```

```
@Inject
```

```
public LiveImagesPart(Composite parent) {  
    public LiveImagesPart(EPartService eps, Composite parent) {  
        eps.addPartListener(this);  
        this.eps = eps;  
    }  
}
```

Bug/Feature #4

- **Description:** On View -> Scene Graph Controls menu, chosen settings always reset to default when you re-open the menu.
- **Steps to reproduce:**
 1. Go to View -> Scene Graph Controls menu
 2. Enable or disable any of the options
 3. Close the window
- **Solution:** My solution was to store the states of the toggles in the FreeFlyerScenario class and implement the functionality to control these toggles from SceneGraphControlsDialog. Now, if a particular checkbox is marked as checked by the user, SceneGraphControlsDialog tells the scenario that it is now checked and to keep it checked until the user unchecks it.



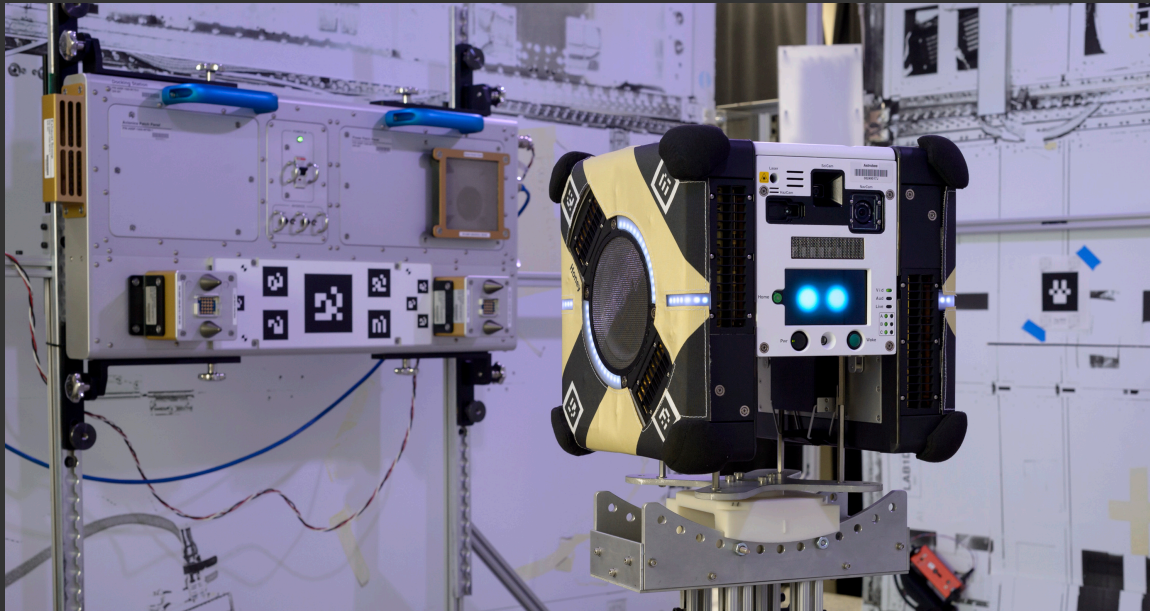
```
private ArrayList<Boolean> FOVCamerasShowing = new ArrayList<Boolean>();
private boolean textShowing = true;
private boolean planTraceShowing = true;

private void initializeCamToggles() {
    int size = robotsList.get(0).getFOVCameraNames().length; // make room for camera names
    for (int i = 0; i < size; i++) { // initialize boolean array to keep track of cameras showing
        FOVCamerasShowing.add(false);
    }
}

// set state of checkbox to what it previously was
if (scenario.textShowing()) {
    text.setSelection(true);
} else {
    text.setSelection(false);
}
```

Lessons Learned and Handoff

- Documentation is important
 - Code can be cryptic
- Familiarizing yourself with the tools you have available is vital
 - Java
 - Eclipse
- Its usually a good idea to make sure you fully understand something before changing it



Acknowledgements and References

Acknowledgements: Special thanks to Jose Benavides, Jonathan Barlow, Katie Hamilton, Roberto Carlino, Ruben Garcia, and the entire Facility team for all the help and guidance.

- <https://www.nasa.gov/astrobee>
- <https://www.nasa.gov/ames/image-feature/nasa-s-astrobee-team-teleworks-runs-robot-in-space>
- https://github.com/nasa/astrobee_gds
- <https://www.nasa.gov/content/spheresastrobee-working-group>

Questions

