ASTROBEE: A NEW PLATFORM FOR FREE-FLYING ROBOTICS ON THE INTERNATIONAL SPACE STATION

Trey Smith[1], Jonathan Barlow[1,2], Maria Bualat[1], Terrence Fong[1],
Christopher Provencher[1,2], Hugo Sanchez[1], Ernest Smith[1,3], and the Astrobee
Team

[1]*Intelligent Robotics Group, NASA Ames Research Center, Moffett Field, CA, 94035*

[2]*Stinger-Ghaffarian Technologies, Inc.*

[3]*Universities Space Research Association*

E-mails*: trey.smith@nasa.gov, jonathan.s.barlow@nasa.gov, maria.g.bualat@nasa.gov,
terry.fong@nasa.gov, chris.provencher@nasa.gov, hugo.sanchez@nasa.gov, ernest.e.smith@nasa.gov*

ABSTRACT

The Astrobees are next-generation free-flying robots that will operate in the interior of the International Space Station (ISS). Their primary purpose is to provide a flexible platform for research on zero-g free-flying robotics, with the ability to carry a wide variety of future research payloads and guest science software. They will also serve utility functions: as free-flying cameras to record video of astronaut activities, and as mobile sensor platforms to conduct surveys of the ISS.

The Astrobee system includes two robots, a docking station, and a ground data system (GDS). It is developed by the Human Exploration Telerobotics 2 (HET-2) Project, which began in Oct. 2014, and will deliver the Astrobees for launch to ISS in 2017.

This paper covers selected aspects of the Astrobee design, focusing on capabilities relevant to potential users of the platform.

1   INTRODUCTION

The Astrobee system draws on AERCam [1] and PSA [2] for small space free-flyer inspiration. It is an upgrade to the SPHERES Research Facility already on ISS. The SPHERES satellites are among the most used ISS payloads, and have hosted diverse experiments ranging from formation flying [3], to satellite inspection [4], magnetic propulsion [5], and visual navigation [6]. However, opportunities to run SPHERES activities are limited, because they require
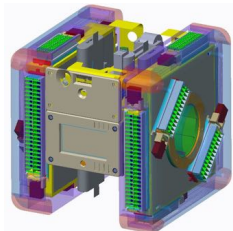

Figure 1: Free flyer front view

substantial astronaut effort for unstowing, setup, replacing consumables, and continuous supervision.

In contrast, the Astrobees have no consumables other than rechargeable batteries, they dock, recharge, and undock autonomously, and they perform most operations without astronaut supervision. Astrobees can also fly farther and faster, have better sensing and computing, and have new capabilities to support research on manipulation and human-robot interaction.

Each Astrobee robot is shaped as a cube 30.5 cm on a side, with mass ~6 kg, and uses battery-powered fans for propulsion (Fig. 1).

2   CONCEPT OF OPERATIONS

Astrobee's concept of operations (conops) was introduced in an earlier paper [7]. Here, we provide a brief review to help motivate the system design.

Astrobees can move throughout the U.S. Orbital Segment (USOS) of the ISS. Each sortie begins and ends with the Astrobee docked at its charging station; docking and undocking are executed autonomously. Operators use a planning tool in the GDS control station software to generate a sortie plan, which contains a mix of 6-degree of freedom (DOF) motion paths and other commands. The robot executes the plan autonomously. During execution, an operator on the ground monitors progress through the control station, which receives video and telemetry feeds. Most faults cause the robot to stop and wait for operator assistance.

There are several alternative ops workflows. For example, during frequent loss-of-signal periods for the ISS space-to-ground link, the robot can continue execution without operator oversight; significant anomalies will pause execution until communications are restored. Operators can also interactively teleoperate the robot, using the control station to send interactive motions and commands similar to those found in plans. The ISS crew can also run the GDS control station software on their laptops, if desired.

Next, we review three of the representative ops

scenarios introduced in [7], with an eye to how they drive the Astrobee system design.

Research Scenario: *A guest scientist uses an Astrobee robot to perform an experiment that requires mounting a custom payload and flying within a specific ISS module.*

Minimizing burden on the ISS crew is an important Astrobee goal. Installing and removing a payload unavoidably requires crew, but crew should not need to be involved in other aspects of experiment setup and teardown, nor should they need to supervise the experiment itself.

This scenario motivates these capabilities: (1) Autonomously dock and undock from the charging station. (2) Follow a plan to navigate to any location on ISS (to pre-position the Astrobee in the module for the experiment). (3) Perch on an ISS handrail (to wait for astronaut assistance, reducing power consumption and staying out of the way of other crew activities). (4) Quick-release payload installation/removal. (5) Execute guest software (to run the experiment). (6) Improve all aspects of mobility performance (max acceleration, max velocity, accuracy, etc.) to allow the broadest possible set of experiments.

Camera Scenario: *ISS Flight Operations uses Astrobee to stream live HD video of a crew activity.*

This scenario motivates these capabilities: (1) Multiple-hour sortie durations (to record long crew activities without interruption). (2) Interactive teleoperation (to support live repositioning when needed to improve the view). (3) Repoint camera while perching (capturing useful video angles with reduced power consumption). (4) End-to-end computing, communications, and GDS capacity to compress, transmit, and view HD video on the ground.

Search Scenario: *ISS Flight Operations uses an Astrobee carrying an RFID reader to confirm the location of a tool needed for an upcoming crew activity.*

This scenario is an example of using an Astrobee to perform a sensor survey; in this case, the RFID reader is the sensor, and we assume the tool is tagged with an RFID. The scenario envisions starting by scanning the module where the tool was last seen, then executing a search algorithm that homes in on the tool by analyzing variations in the RFID signal strength.

This scenario motivates these capabilities: (1) Build and execute extensive survey plans that include many target positions and commands to take sensor samples. (2) Minimize oversight burden on operator (survey activities can be long and monotonous). (3) Execute guest software (search algorithm provided by RFID experts).

3 ROBOT DESIGN

Each Astrobee robot is cube-shaped, ~30.5 cm on a side, weight ~6 kg (Fig. 1). Although Astrobees can move in any direction, they have a preferred forward motion direction and a top side that nominally faces the overhead or "ceiling" of the ISS interior.

Prominent features include: the propulsion modules that occupy the entire left and right faces of the cube, the touch screen and a variety of sensors on the forward face, the dock adapter and perching arm on the aft face, and open payload bays at bottom center.

3.1 Propulsion

The Astrobees are designed to operate in the pressurized intra-vehicular (IV) environment but not the extra-vehicular (EV) environment of the ISS. Early in the Astrobee concept development process, we studied producing a dual-use robot that could operate both IV and EV, for example, using cold-gas propulsion with autonomous refueling. We rejected this option because ensuring EV safety would have greatly increased project costs and imposed design constraints that would have reduced the robots' IV capability.

The selected IV-only propulsion design uses battery-powered fans. With electricity as the only consumable, we can support autonomous resupply with relatively simple autonomous docking to recharge.

Each Astrobee has two propulsion modules that occupy the entire left and right sides of the robot. Each module is built around a plenum; the plenum is pressurized by a single centrifugal impeller, which draws in air through a central intake and feeds six exhaust nozzles around the edges.

The two propulsion modules are interchangeable. They are arranged so that their impellers rotate in opposite directions and generally at the same speed, canceling unwanted drag torque and gyroscopic moment disturbances. The impeller speed is adjustable to trade between peak performance vs. reduced power and noise. Basing each module on a single central impeller allows us to maximize the impeller diameter, reducing RPM rate, power, and noise. Confining the propulsion system to the left/right sides of the robot, with both intakes and nozzles directed outward, also greatly improves design modularity—the complexities of the central module and guest payloads that fit between the propulsion modules are isolated from the propulsion air flow and can be excluded from aerodynamic analysis and performance optimization.

With a constant impeller speed, the thrust from each nozzle has fixed direction and continuously adjustable magnitude, controlled by adjusting the nozzle open area with a servo that actuates two flappers. The flappers can adjust from fully open to fully closed in <100 ms, and flapper motion introduces negligible disturbance forces. Through careful design and prototype iteration, the nozzles achieve aerodynamic discharge coefficient >0.9 (1 is perfectly efficient) and consistent thrust alignment, even when nearly closed.

A propulsion system is holonomic in 6-DOF if it has the ability to instantaneously apply force in any direction and torque about any axis. This capability is important for Astrobee as a research platform, because

a holonomic thruster arrangement can simulate the thruster arrangement of any other spacecraft of interest. Holonomic control is provided by the 6 x 2 = 12 independent thrusters. The nozzle geometry is arranged with thrust vectors offset, such that they do not pass through the robot's body frame origin (geometric center); thus, each nozzle produces a combination of force and torque. We can construct a pure force on any Cartesian axis (+/-X, Y, or Z) by applying equal thrust with a matched pair of nozzles that have aligned forces and opposing torques. Likewise, we can construct a pure torque with a pair of nozzles that have aligned torques and opposing forces (in fact, each torque axis can be produced with either of two redundant nozzle pairs).
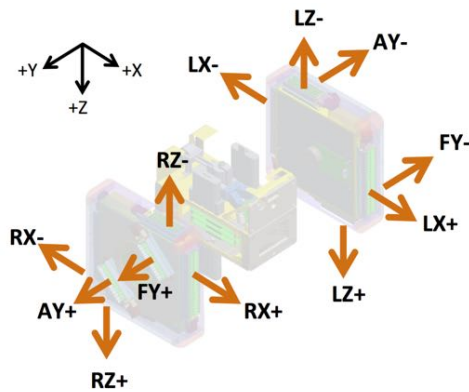

Figure 2: Exploded view with nozzle geometry

It is possible to achieve *n*-DOF holonomic control with as few as $n + 1$ unidirectional thrusters [8], but by using $12 > 7$ thrusters, we improve efficiency and make the force allocation problem better conditioned and more robust to deviations from the calibrated propulsion model (such as slightly misaligned thrusters). In case of certain types of thruster malfunctions, it might be possible to take advantage of redundancy to continue flying an Astrobee with reduced performance [9], but implementing this capability is not in scope for the current project.

Although Astrobees have holonomic control, their maximum acceleration capability is not symmetrical. When following a typical trajectory, flying forward, most of the required thrust is starting and stopping on the forward/aft (X) axis; this axis has the largest nozzles that achieve the full max acceleration requirement (10 cm/s^2). Nozzles on the right/left (Y) and bottom/top (Z) axes are mostly used for turning and for rejecting environmental disturbances; for these axes, we reduced the nozzle size, giving up peak acceleration to save space and improve control precision at lower thrust levels. A further asymmetry comes from load balancing between the two propulsion modules. Pure forces are constructed by using a pair of nozzles. For X and Z forces, the two nozzles are on different propulsion modules, so the

force can draw power from both impellers. Y forces use two nozzles on the same module, drawing power from a single impeller, which reduces the maximum force available.

An important propulsion design driver was the need for efficiency—producing the required thrust with low noise and power consumption. Basic rocket physics dictates that with a fixed fraction of reaction mass, the achievable Δv scales linearly with increasing exhaust velocity v_e, but with the downside that ideal thrust power efficiency (W/N) scales inversely with v_e.

Typical spacecraft design, with limited reaction mass, seeks to maximize Δv by increasing v_e. Astrobees, on the other hand, have unlimited reaction mass (air in the pressurized cabin), allowing us to focus on power efficiency by *minimizing* v_e. Astrobee nozzles operate efficiently at low pressure (~0.1 psi) and exhaust velocity (~11 m/s). For comparison, the SPHERES satellites use EV-analog cold gas thrusters that operate at ~25 psi with exhaust velocity ~250 m/s.

A further advantage of low-pressure propulsion is that the required pressure can be produced by an efficient centrifugal impeller, eliminating any need for a compressor, and precision engineering is not required to adequately seal the plenum.

Astrobee exhaust velocity is lower bounded by the need to achieve sufficient thrust while fitting within volume constraints. The open area of each nozzle ranges from ~40-65 cm^2 (compare to ~0.005 cm^2 for SPHERES), and together they occupy around 10% of the total robot surface area.

All moving parts of the propulsion system are enclosed, which is important for reliability and safety. The intake is covered by a screen with ~1 mm openings to avoid drawing in crew hair or drawing in and ejecting particles floating in the cabin. The nozzle flappers are covered with a coarser grille to protect fingers and keep out objects that could cause jams.

3.2 Structure

Modularity was a key design driver. Astrobee's major modules, such as the propulsion modules, forward and aft panels, and payloads, support straightforward removal and replacement by crew. In some cases, spares will be stocked on orbit to enable repairs. Astrobees use captive fasteners to mitigate the risk of loose fasteners in the crew cabin, so crew maintenance does not require a glove box.

Each Astrobee's core frame is aluminum. The propulsion module plenum is primarily 3D printed using lightweight Windform XT, and saves mass by making dual-use of exterior impact foam as a structural material.

3.3 External Sensors

Each Astrobee uses a suite of six commercial off-the-shelf (COTS) external sensors (Fig. 3). First is the *NavCam*, a forward-facing monocular RGB imager with 130° field of view (FOV), fixed focus, and 1.2

megapixel (MP) resolution. NavCam images feed the sparse mapping and optical flow portions of the general-purpose pose estimator (section 3.4), and are also useful for operator situation awareness due to the wide FOV.

For improved relative pose estimation during docking and perching, Astrobees use *DockCam* and *PerchCam*. DockCam is a clone of NavCam that faces aft, where it can view fiducials on the Astrobee dock during the docking approach. PerchCam is an aft-facing CamBoard Pico Flexx time-of-flight flash LIDAR depth sensor with 62° FOV, 224 x 172 resolution, and effective range 0.1-4 m. It detects ISS handrails based on their 3D shape and determines the Astrobee's relative pose during perching approach.

To detect obstacles, Astrobees use *HazCam*, a clone of PerchCam that faces forward.

To stream video of crew activities, Astrobees use *SciCam*, a forward-facing RGB imager with 54.8° FOV, 13 MP resolution, and auto-focus, similar to the camera module found on the back of many modern smart phones.

Finally, to provide a redundant velocity estimate, Astrobees use *SpeedCam*, a top-facing PixHawk PX4Flow integrated sonar/optical flow sensor, which does its own internal data processing and provides 3D velocity estimates directly to Astrobee's low-level processor.
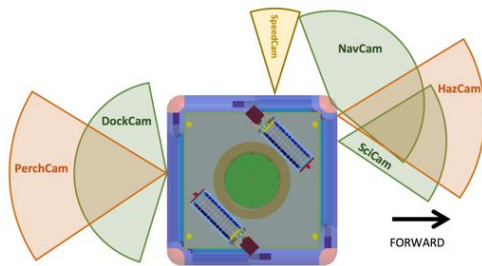

Figure 3: Side view with sensor viewing angles.

3.4 Position Estimation

Astrobee position estimation has four main modes: *general-purpose*, *fiducial-relative*, *perch-relative*, and *impaired*. All modes use an augmented-state extended Kalman filter (EKF) for sensor fusion, and all make use of gyroscope and accelerometer readings from Astrobee's inertial measurement unit (IMU). They differ in the additional inputs used by the EKF.

General-purpose mode is used to estimate position anywhere in the ISS USOS, with accuracy sufficient for reliable navigation in free space (20+ cm away from walls). In this mode, the EKF has two additional inputs. First, visual texture features on the ISS walls observed in NavCam images are matched with a prior *sparse map* of features, providing an absolute position update at ~2 Hz. Second, an optical flow algorithm matches features across frames in the same image

stream at ~6 Hz, providing smoother relative updates.

Fiducial-relative mode provides more accurate and robust position information when artificial fiducials (or "AR targets") are in view. It matches fiducials detected in NavCam or DockCam images with a prior fiducial map. It will be used in two main scenarios. First, for autonomous docking, in order to achieve the ~1 cm position accuracy needed for successful mating, the Astrobee dock will have fiducials. Second, in order to support guest science needs for improved motion accuracy during experiments, we may outfit a designated workspace in the ISS with several fiducials on the surrounding walls.

Perch-relative mode provides accurate pose estimates relative to an ISS handrail, based on 3D point clouds from the PerchCam depth sensor. It provides the ~2 cm position accuracy required near the end of the perching approach for successful grasping. The perch estimator fits a handrail geometric model to a point cloud, recognizing the handrail's linear shape and relationship to the planar wall behind it.

Impaired mode is used when inputs required for the other modes are not available, for example due to a sensor fault or software fault on the mid-level processor. Impaired mode uses the 3D velocity estimate provided by the SpeedCam, which allows the Astrobee to stop and station keep until the fault is resolved.

More detailed papers on the general-purpose and perch-relative estimators will appear soon.

3.5 Navigation and Control

Astrobee's navigation stack executes 6-DOF motion trajectories. These trajectories are generated by the GDS control station planner or by Astrobee's flight software. In either case, the trajectories are constructed to ramp velocity smoothly and respect operating limits such as max velocity and acceleration constraints. A trajectory is represented as a spline curve of polynomial *chunks* parameterized by time.

The navigation and control pipeline operates as follows: (1) *Navigator:* Stores the overall trajectory, monitors progress, and feeds new chunks into the pipeline as needed. (2) *Command shaper:* Taking into account any deviation from the target trajectory, calculates a short-term repair trajectory that smoothly transitions from the current state (12-DOF pose+velocity) to rejoining the target trajectory, while respecting operating limits. (3) *Controller:* A PID controller that calculates a force/torque command designed to zero the error between the current state and the command shaper's repair trajectory. (4) *Force allocator:* Calculates the nozzle servo positions that implement the controller's force/torque command. (5) *Output:* The nozzle command from the force allocator is forwarded to the propulsion module controller via I2C, where it is translated to PWM outputs for the individual servos.

While most of the control approach is

straightforward, there are some particular challenges. First, it is non-trivial to calibrate the propulsion physics model needed by the force allocator. The thrust output by any nozzle has a non-linear dependence on both the RPM rate of the impeller (which increases plenum pressure) and the open area of every other nozzle (which decreases pressure). Second, Astrobee configuration changes, such as moving the arm and adding or removing payloads (which may include integrated thermal fans), will induce inertial property changes and new disturbance forces. Finally, during activities like perching and docking, the controller will need to accommodate contact forces.

3.6 Computing and Software Architecture

An Astrobee has three main processors: (1) The *Low-Level Processor* (LLP) controls the propulsion system, and reliably runs the high-rate control software, in isolation from the rest of the software. (2) The *Mid-Level Processor* (MLP) runs most of the software developed by the Astrobee team, including the compute-intensive machine vision algorithms, in isolation from guest software. (3) The *High-Level Processor* (HLP) is primarily for guest science use, but also hosts less critical functions such as streaming HD video compression and touch screen interaction.

To achieve a good combination of power, size, and performance, we use COTS ARM architecture processors provided in a system-on-module format, derived from modern smart phone technology. The LLP is a Wandboard Dual with two cores that run at up to 1 GHz and max power ~3 W. Both the MLP and HLP are IFC6501 modules, each with four cores that run at up to 2.5 GHz and max power ~10 W. The three main processors ride on two custom carrier boards that are designed to enable a possible on-orbit upgrade by crew later in the project life cycle. Each Astrobee also carries PIC microcontrollers for power management and distributed control of the propulsion modules and perching arm.

The three main processors communicate with each other and with the dock via Ethernet. Other Astrobee avionics components and payloads mostly communicate using USB 2.0 or I2C data buses.

Astrobee flight software uses a service-oriented robotic architecture style [10], built on the Robot Operating System (ROS) middleware. The LLP and MLP run Ubuntu Linux, and the HLP runs Android, which drives the touch screen display.

3.7 Power

Astrobees carry rechargeable Li-ion batteries, specifically Inspired Energy model ND2040HD34. An Astrobee can operate with as few as two batteries (reducing mass and improving performance) or as many as four (increasing battery life). Each battery outputs 14.4V nominal, stores 49 Wh of energy, and incorporates numerous safeguards to avoid the risk of fire or leakage in the ISS environment.

Each Astrobee has an electrical power system (EPS) board based on a PIC microcontroller, which manages power consumption by switching components on only when needed, provides the ability to boot and reset the main processors, and outputs telemetry on internal temperatures and power system health.

3.8 Communications

During sorties, Astrobees communicate commands, telemetry, and video through the ISS WiFi network and Ku-band downlink. After sorties, Astrobees can transfer large files through an Ethernet connection with the dock.

Commands and telemetry use NASA's Robot API Delegate (RAPID) message conventions and are delivered by RTI Data Distribution Service (DDS) middleware, which offers tunable quality-of-service that can tolerate high-latency networks and dropped packets. HD video is H.264 compressed and streamed to the ground via the Real-Time Transport Protocol (RTP).

Because the WiFi and Ku-band networks are subject to frequent dropouts, Astrobees keep onboard logs of all streaming data, allowing later downlink of a pristine copy.

3.9 Thermal

Engineers are accustomed to relying on passive gravity-driven thermal convection to provide free cooling; reduced convection in the zero-g ISS environment makes careful thermal design especially important.

The Astrobee core module is cooled using a pair of always-on cross-flow thermal fans that draw in air from forward intake vents, blow it over a heat sink that cools the MLP and HLP, and exhaust aft. These fans do produce a net disturbance force, but it is constant and small (<5% of propulsion system thrust).

The propulsion module impeller and servo motors are the other main heat sources. Since these motors only operate when the propulsion system is active, they are naturally cooled by the propulsion air flow.

A variety of anomalies could cause overheating: jammed motors, blocked vents, failed fans, etc. In these situations, over-temperature power cutoffs ensure thermal safety.

3.10 Perching Arm

Each Astrobee carries a perching arm on its aft side that allows it to grasp ISS handrails and dwell for extended periods, reducing power consumption and interference with ISS operations [11-12]. It could also support future manipulation research.

The arm has three degrees of freedom: two joints and a gripper. The joints allow the arm to stow completely within Astrobee's protective foam exterior when not in use, and deploy only for grasping.

The gripper is an underactuated tendon-driven

design with three fingers. When unpowered, it is passively spring-loaded shut; winding up the tendon opens all three fingers.

Once an Astrobee grasps a handrail, it powers down its propulsion system, and its arm joints double as a pan-tilt unit for pointing cameras located on the opposite (forward) side of the robot (Fig. 4).

The arm is designed to be flexible and backdrivable; its grip is not strong enough to injure the crew, and a perched Astrobee can be easily removed from a handrail if it is in the way in an emergency. The crew can also manually backdrive and perch the arm on a handrail, as a convenient way to move an Astrobee when needed.
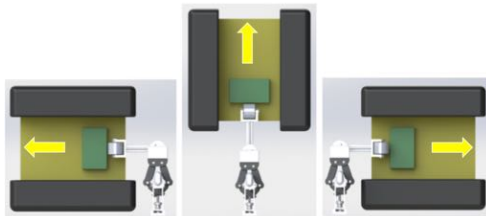

Figure 4: The perching arm grasps an ISS handrail and doubles as a pan/tilt unit

3.11 Collisions

Astrobee robots must not injure crew or damage the ISS. As a free-flying robot in a tightly confined space, collisions are a particular concern.

For comparison, the SPHERES satellites handle collision safety primarily by assuming the constant presence and oversight of crew. The Astrobee conops, on the other hand, calls for performing most sorties end-to-end without crew tending, except in case of anomalies.

The Astrobees use a tiered approach to mitigate collisions. First, to ensure crew and ISS safety, Astrobee hardware is designed to be inherently collision-safe, regardless of any software anomalies. Second, to minimize ops disruptions, Astrobee flight software includes several collision mitigation features.

We ensure collision safety by (1) hardware limiting the maximum thrust capability of the propulsion system to 0.72 N, and (2) encasing each Astrobee in an impact-absorbing foam shell, similar to a bicycle helmet, with rigid hardware recessed behind the shell. The worst-case kinetic energy of a collision is approximately the propulsion force times the longest straight-line path in the ISS interior (~21 m). Factoring in air resistance, the maximum impact velocity is ~2.1 m/s, equivalent to a 23 cm drop in Earth surface gravity. Impacts near the worst-case velocity will crush the foam shell by up to ~1.5 cm depth, absorbing most of the impact energy in the process.

Astrobee flight software has several features that mitigate collisions. The simplest is an over-speed cutoff: when an Astrobee detects that it has exceeded its nominal maximum speed of 0.5 m/s, or loses its velocity fix, it powers off the propulsion system. This feature limits impact velocity to below 0.5 m/s, reducing the equivalent drop height to 1.3 cm.

To further avoid low-speed collisions, we combine multiple layers of checks: (1) Astrobee trajectories will generally be specified by a trained operator, with an opportunity to view a 3D simulation of flying the trajectory in the ISS environment. Using this feature, operators will catch and correct some errors prior to execution. (2) Astrobees will store a list of known keepout zones, such as module walls, projecting hardware, and areas near exhaust vents that could blow them off course. An Astrobee will refuse to execute a trajectory that goes too close to a keepout zone. (3) The HazCam can detect unexpected obstacles, causing the Astrobee to stop and request operator assistance.

We note that, despite these measures, like humans in crowded office environments, Astrobees operating unattended in the complex and dynamic ISS environment will unavoidably bump into objects from time to time, with minimal consequences. Astrobee's safety features primarily focus on preventing higher-speed collisions and effectively controlling the risk of crew injury or damage.

3.12 Human-Robot Interaction

The Astrobees are designed to interact with people, including the ISS crew, flight controllers, and the general public (for example, during crew outreach videos). They will also be used for human-robot interaction research.

Each Astrobee carries a touch screen, speaker and microphone, signal lights, and a laser pointer. Initially, the Astrobees will use these components primarily to help crew understand the Astrobee's state and intentions (for example, providing turn signals). Eventually, they could also provide capabilities like video telepresence for flight controllers, and pointing the laser to communicate a precise location to a crew member.

3.13 Dock

The Astrobee docking station has two berths, each providing power and Ethernet connectivity to one Astrobee (Fig. 5).

When docking, an Astrobee autonomously approaches its berth using visual servoing relative to fiducials mounted to the dock. Once contact is made, a system of conical lances (on the berth) and cups (on the robot) guides the final mating, accommodating up to ~1 cm of alignment error. The berth connector has 20 spring-loaded pogo pins that contact matching pads on the robot side. Compliance in the pins accommodates any remaining alignment error.

When mating is complete, permanent magnets on

the berth attract striker plates on the robot, providing passive retention force. To enable undocking, push rods extend from the dock and push the Astrobee ~3 cm away, sufficient separation for the propulsion system to easily overcome the reduced magnetic force.

The dock also includes COTS battery chargers sufficient to simultaneously charge a full set of four stand-alone Astrobee batteries. This function provides spare charged batteries, allowing an Astrobee to execute back-to-back activities when crew are available to assist with battery swapping.
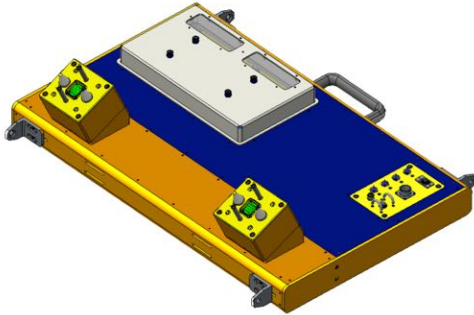
Figure 5: Docking station with two berths. Push rods shown extended, fiducials not shown.
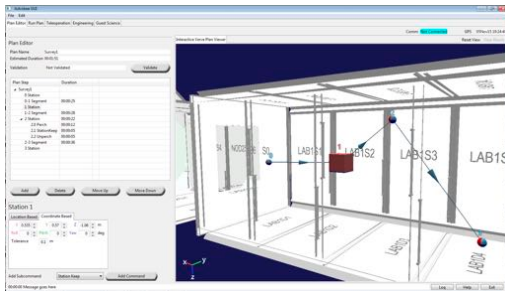

Figure 6: Early version of GDS control station planning interface

3.14 Ground Data System (GDS)

Operators monitor and command Astrobee through control station software. Control stations can be used by ISS flight controllers, research facility staff, or guest science researchers at their own institutions.

The control station user interface is implemented using the Eclipse Rich Client Platform (RCP) toolkit. It provides tools for planning (Fig. 6), execution monitoring, and live teleoperation. As the control station receives position telemetry, it renders the robot's estimated pose within a 3D model of the ISS, in combination with video and status displays.

The GDS also includes servers for archiving Astrobee data, and a suite of engineering software tools to support ongoing maintenance and upgrades of the Astrobee robots.

4 GUEST SCIENCE

Astrobee guest science will build on the successful SPHERES Research Facility. Guest scientists will pursue their own funding to develop their experiments, and receive in-kind support from the new Astrobee Research Facility (integration guidance, ground testing facilities, etc.) and the ISS program (launch services, etc.).

The SPHERES Working Group (SWG) is a community of researchers interested in using the SPHERES facility. The Astrobee team is working with the SWG to develop plans for the transition to Astrobee, and NASA has already awarded early funding to develop Astrobee-relevant payloads, through the Small Business Innovative Research (SBIR) and Early Stage Innovation (ESI) programs.

Beyond the research areas initiated with SPHERES, future Astrobee research is likely to include: (1) *Free-flyer manipulation*, employing the Astrobee perching arm, or adding advanced manipulator payloads [13]; (2) *Free-flyer human-robot interaction* (HRI) using the Astrobee touch screen, speaker, microphone, laser pointer, and signal lights, or adding new hardware [14].

Researchers can modify the Astrobee system in a variety of ways. The simplest is to add an onboard software node that interacts with the robot using high-level APIs defined through the Robot Operating System (ROS). Guest software can subscribe to telemetry and send commands such as robot motions, arm motions, and HRI interaction, including providing an astronaut user interface through the touch screen. The robot's Android/ARM high-level processor is set aside to host guest software, and isolated from high-rate control code.

If guest software needs functions not provided in the high-level API, such as experimental control algorithms that need to run at high rates, researchers can customize the Astrobee core software for their experiment. We plan to open source release the core software code base to facilitate community contributions. Note that core software changes will need careful review to ensure reliability.

Guest scientists can also add hardware payloads to the robot. Each Astrobee can host up to three simultaneous payloads, each within a ~15 x 15 x 10 cm peripheral bay, attached with a quick-release mechanism: two tool-free quarter-turn fasteners whose closure also engages the blind-mate connector that carries power and USB 2.0 data.

The peripheral bays are arranged with one on top (aft) and two on the bottom (forward and aft). The bays are recessed behind the foam shell that absorbs impact energy in a collision. The two bottom bays are contiguous, so an Astrobee can accommodate one double-size payload. The top bay is often occupied by the Astrobee's perching arm, which can be swapped with a payload as needed.

Unlike aircraft in 1g environments, an Astrobee's

maximum payload is not limited by its lifting power. Much larger payloads could attach to an Astrobee's aft dock adapter in a tractor-trailer configuration, or multiple Astrobees could cooperate to exert stronger forces on a payload [15]. Note that if payloads extend outside the peripheral bays or exceed ~1 kg mass, further analysis would be needed to ensure collision safety and update control software.

As a pilot for demonstrating Astrobee payload and survey capabilities, we are coordinating with the RFID-Enabled Automated Logistics Management (REALM) project at Johnson Space Center [16]. Astrobee will serve as a mobile RFID reader platform to search for missing items, complementing fixed readers in cabinets and hatches. The REALM Astrobee payload will use a bottom payload bay, together with thin flat flexible antennas mounted on the propulsion module faces.
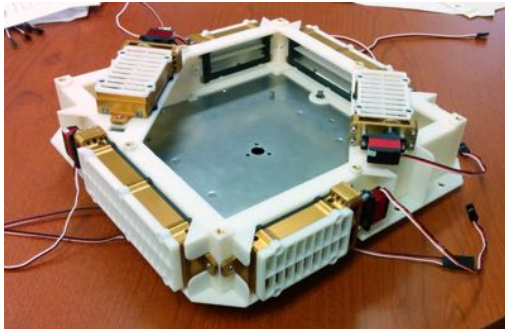

Figure 7: P4 propulsion module fit check, prior to integration of impeller and foam shell

5 PROJECT STATUS

The Astrobee project is currently performing ground testing of Prototype 4 (P4), the fourth in a series of integrated risk reduction prototypes. P4 is the first prototype to have flight-like size, structure, and propulsion (Fig. 7), and include a full docking station.

Lessons learned from P4 will be merged into the final design iteration, prior to verification of a certification unit later this year.

6 CONCLUSION

Astrobee is a flexible platform that will enable the next generation of zero-g free-flying robotics research.

References

[1] Frederickson S, et al. (2003) Mini AERCam: Development of a free-flying nanosatellite inspection robot. In *Proc. SPIE 5088 Space Sys. Tech. Oper*.

[2] Dorais G A, et al. (2003) The Personal Satellite Assistant: An internal spacecraft autonomous mobile monitor. In *Proc. IEEE Aerospace*.

[3] Ramirez-Riberos J L, et al. (2010) Distributed control of spacecraft formations via cyclic pursuit: Theory and experiments. *J. Guid. Control Dynam.*, 33(5): 1665-1669.

[4] Tweddle B E, et al. (2012) The SPHERES VERTIGO Goggles: Vision based mapping and localization onboard the International Space Station. In *Proc. iSAIRAS*.

[5] Porter A, et al. (2014) Demonstration of electromagnetic formation flight and wireless power transfer." *J. Space Rob.*, 51(6): 1914-1923.

[6] Fong T, et al. (2013) Smart SPHERES: A telerobotic free-flyer for intravehicular activities in space. In *Proc. AIAA Space*.

[7] Bualat M, et al. (2015) Astrobee: Developing a free-flying robot for the International Space Station. In *Proc. AIAA Space Forum*.

[8] Davis C (1954) Theory of positive linear dependence. *Amer. J. Math*. 74:4 (Oct. 1954), 733–746.

[9] Pong C M, et al. (2011) Autonomous thruster failure recovery on underactuated spacecraft using model predictive control. In *Proc. AAS Guid. Control Conf.*

[10] Flückiger L and Utz H (2014) Service oriented robotic architecture for space robotics: Design, testing, and lessons learned. *J. Field Rob*. 31(1): 176-191.

[11] Yoo J W, et al. (2015) Avionics and perching systems of free-flying robots for the International Space Station. In *Proc. IEEE Int. Symp. Systems Eng*.

[12] Park I W, et al. (2015) Developing a 3-DOF compliant perching arm for a free-flying robot on the International Space Station. In *Proc. ICRA*.

[13] Estrada M A, et al. (2016) Free-flyer acquisition of spinning objects with gecko-inspired adhesives. In *Proc. ICRA*.

[14] Szafir D, et al. (2015) Communicating directionality in flying robots. In *Proc. HRI*.

[15] Lapilli G, et al. (2015) Results of microgravity fluid dynamics captured with the SPHERES-Slosh experiment. In *Proc. IAC*.

[16] Broyan J L, et al. (2014) Logistics reduction technologies for exploration missions. In *Proc. AIAA Space*.