IAC-23,D1,6,1,x78669

Multi-Agent 3D Map Reconstruction and Change Detection in Microgravity with Free-Flying Robots

Holly Dinkel[1][§], **Julia Di**[2][§], **Jamie Santos**[3], **Keenan Albee**[4], **Paulo Borges**[5],
Marina Moreira[6], **Oleg Alexandrov**[6], **Brian Coltin**[6], and **Trey Smith**[6]

[1]*Department of Aerospace Engineering at the University of Illinois Urbana-Champaign, Urbana, IL, USA*
[2]*Department of Mechanical Engineering at Stanford University, Stanford, CA, USA*
[3]*Department of Physics at Chalmers University of Technology, Gothenburg, Sweden*
[4]*Maritime and Multi-Agent Autonomy Group at the Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, USA*
[5]*Robotics and Autonomous Systems Group at CSIRO Data61, Brisbane, QLD, Australia*
[6]*Intelligent Robotics Group at the NASA Ames Research Center, Moffett Field, CA, USA*

Abstract

Assistive free-flyer robots autonomously caring for future crewed outposts—such as NASA's Astrobee robots on the International Space Station (ISS)—must be able to detect day-to-day interior changes to track inventory, detect and diagnose faults, and monitor the outpost status. This work presents a framework for multi-agent cooperative mapping and change detection to enable robotic maintenance of space outposts. One agent is used to reconstruct a 3D model of the environment from sequences of images and corresponding depth information. Another agent is used to periodically scan the environment for inconsistencies against the 3D model. Change detection is validated after completing the surveys using real image and pose data collected by Astrobee robots in a ground testing environment and from microgravity aboard the ISS. This work outlines the objectives, requirements, and algorithmic modules for the multi-agent reconstruction system, including recommendations for its use by assistive free-flyers aboard future microgravity outposts.

Keywords: Change Detection, Microgravity, Robotics, 3D Reconstruction, Mapping, Artemis

1. Introduction

On the International Space Station (ISS), astronaut time is expensive and limited. Future orbital outposts, including Gateway, will likely have limited crews or even be uncrewed for extended periods of time. Routine tasks on these outposts provide compelling opportunities for robotic assistance [9] [17]. A number of free-flying assistants have been proposed to address this need including the SPHERES [24], CIMON, Int-Ball [4], and most recently Astrobee [31] robots. Astrobee is a next-generation free-flying robot onboard the ISS capable of deploying a variety of future research payloads and guest science software intended to serve as an astronaut assistant to alleviate valuable crew time [31, 6, 5, 21]. One area where Astrobee has the potential to reduce astronaut workload is as a mobile sensor conducting and recording surveys. Astronauts currently conduct manual regular visual surveys

of the entire space station, which Astrobee could alleviate. Free-flying robots, including Astrobee, will be increasingly useful assistants, managing and maintaining future microgravity outposts.

However, one challenge for deploying robots in human habitats is the interiors of these environments are ever-changing: objects frequently disappear or are reintroduced. A current map of the environment is vital for maintenance and navigation, but creating a new map and computing between-map changes is resource-intensive as robots could take hours to complete an environment scan. Rather than repeat the mapping process for the entire environment, it is advantageous to only recompute local changes—Fast Image-Based Geometric Change Detection (FastCD) enables robots to quickly locate changes in a known world model based on geometric projection operations performed on a small (< 10) batch of images as shown in Figure 1 [25]. This work develops FastCD for operation on assistive free-

[§]Equal contribution
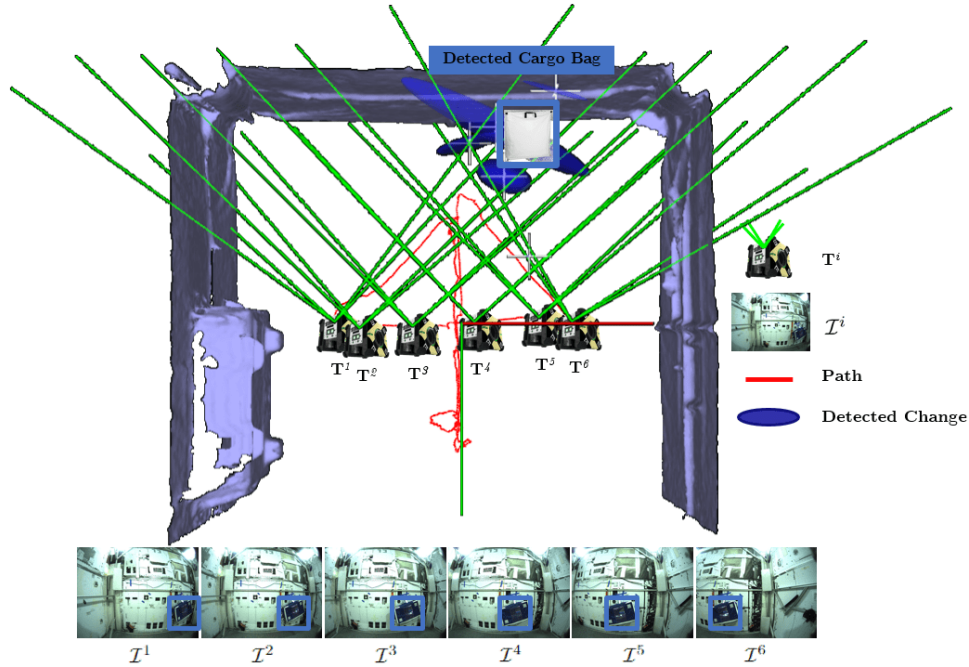
Figure 1: FastCD detects scene changes against a pre-built 3D model of the environment by computing inconsistencies among a set of RGB images projected onto the 3D model using their corresponding poses, $\mathbf{T}^i$.

flyers in microgravity. This work contributes:

1. Discussion of a multi-agent scene change detection framework for free-flyer robots. This application can also be extended for object discovery.
2. Discussion of scene change detection considerations for resource-constrained mobile robots in a space environment.
3. Demonstration and evaluation of FastCD on data collected with Astrobee robots at the NASA Ames Research Center Granite Lab microgravity research facility and aboard the ISS.
4. Release of a FastCD-compatible dataset of Granite Lab and ISS activity data at https://bit.ly/astrobee_fastcd_data.

2. Related Work

Because environment models should be up-to-date for most applications, a common task in robotic surveying is environment reconstruction. For larger environments, obtaining a new scan and generating 3D models of the environment can be expensive, requiring dedicated robot mapping time, sensors, and significant data processing. This cost can be reduced through 3D scene change detection [27]. By identifying only local changes

in an environment based on existing information, which can be generated directly from raw imagery, a mobile robot may direct its surveying to target change regions.

There are several types of change detection methods categorized according to the target scene. Classical change detection is commonly formulated as a 2D or 2.5D problem, comparing new image data to old background images, and is adopted in several satellite remote sensing and surveillance use cases [1, 8, 28]. Pairwise image comparison is straightforward, but may not scale proportionally to map size. For traversing through a 3D environment in robotic outpost maintenance, a 3D change detection approach is better suited.

Within 3D change detection, several works have investigated probabilistic modeling. One recent unsupervised method of scene change detection uses Gaussian Mixture Model (GMM) Clustering to identify changes between complete point cloud maps of the environment [30]. This work processes a complete point cloud map from one survey into region-based GMM clusters through split-and-merge expectation-maximization [18], and detects between-scene changes by comparing the cluster sets with the Earth Mover's Distance [29]. Since this method requires complete point cloud maps of the environment, it cannot be

deployed online on a mobile platform and is better suited for situational awareness applications rather than anomaly response.

Another recent method repurposed map-based localization for change detection [15]. This method does not perform well if self-localization methods are poor, which may be the case in a space environment. For example, free-flying assistive robots in microgravity do not have the availability of a gravity vector for localization. Furthermore, in dense environments like the ISS, map landmarks are not always available when there are occlusions from cargo bags or human activity. Maps are also expensive to build online for a mobile robot with limited computing power.

Other hybrid methods augment a 2D method with 3D information for remote sensing applications. Changes are detected by comparing unordered 2D images collected by a mobile agent to a 3D building information model which contains information about the geometry of the environment [14]. This method may work well for construction applications where environments are developed according to a blueprint, but is not well suited for change detection in human habitats which are frequently significantly reconfigured and for which a blueprint may be unavailable.

Finally, other geometric treatments of 3D change detection create inconsistency maps from RGB images or depth projections when compared to 3D models [25, 35, 36]. These methods are fast and lightweight, suitable for operation on a mobile robot when given a 3D world model of the environment, but are sensitive to localization accuracy.

3. Methodology

This work applies FastCD to sequences of RGB images to detect scene changes and discover novel objects. The FastCD system is illustrated in Figure 2 and discussed in the following sections. First, raw image and pose data are collected with the Astrobee robot (Section 3.1) and processed to remove low-movement images (Section 3.2). A camera pose is estimated for each image (Section 3.3). A baseline 3D model of the environment is created (Section 3.4). Finally, candidate inconsistencies are computed between the images used for change detection, and the resulting 3D regions of change are output (Section 3.5).
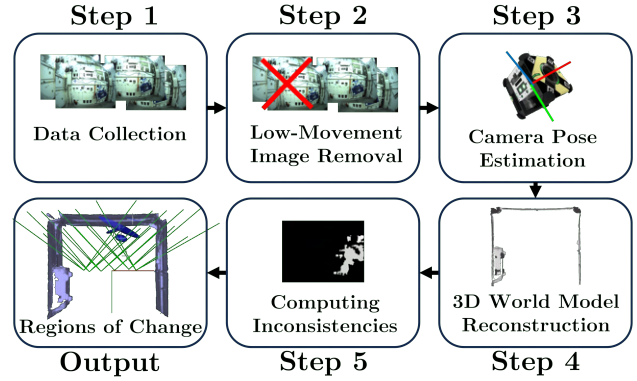
Figure 2: From an image sequence, the change detection system removes low-movement images, estimates the camera pose for remaining images, builds a 3D environment map, computes between-image inconsistencies, and outputs detected changes.
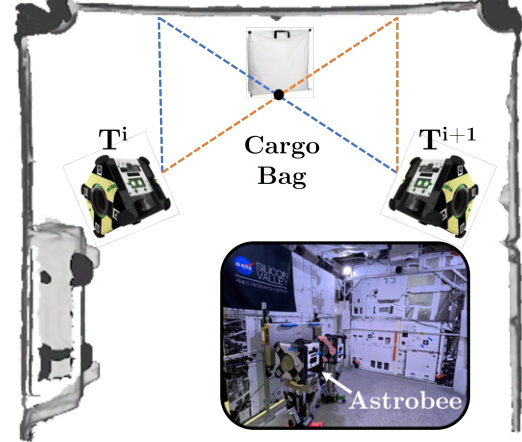
Figure 3: FastCD first back-projects images captured from different poses, $\mathbf{T}^i$ and $\mathbf{T}^{i+1}$, onto a 3D model. Next, it re-projects the 3D points from the model onto a common image plane for comparison. (Inset) The Granite Lab is an environment which mimics microgravity for testing the ground Astrobee units.

Figure 3 provides geometric intuition for FastCD, summarized in the following steps:

1. Let i represent the index of an image-pose pair in the data sequence.
2. Back-project (pixel-to-point) image $\mathcal{I}^i$ from pose $\mathbf{T}^i$ into the World frame of the 3D model using the pose and projection matrix of the camera.

3. Re-project (point-to-pixel) image $\mathcal{I}^i$ onto the image plane $\mathcal{I}^{i \to i+1}$ of the camera at pose $\mathbf{T}^{i+1}$ to create a new image containing the content from the image captured at $\mathbf{T}^i$ at pose $\mathbf{T}^{i+1}$.

If there is no change in the 3D model between when the image was acquired and when the model was created, all pixels taken from pose $\mathbf{T}^i$ should correctly re-project onto $\mathcal{I}^{i \to i+1}$. In other words, the image $\mathcal{I}^{i+1}$ at $\mathbf{T}^{i+1}$ and the image $\mathcal{I}^i$ from $\mathbf{T}^i$ re-projected onto $\mathcal{I}^{i \to i+1}$ should be approximately identical. If there is a change from what is conveyed in the model, pixels corresponding to the change will re-project onto $\mathcal{I}^{i \to i+1}$ incorrectly.

3.1 Data Collection

Data in this work were collected with Astrobee units in the Granite Lab and on the ISS. The Granite Lab (inset in Figure 3) is a facility that replicates visual features of the ISS and mimics 3 DOF microgravity by placing Astrobee on a near-frictionless air bearing.

The Navigation Camera (NavCam) on Astrobee is a fixed-focus RGB camera with a wide field of view. This sensor collects Bayer images (at 1280×960 resolution) at 5 Hz [31, 21]. When Astrobee surveys its environment, it flies to waypoint stations where it may hold its pose to capture clear, high-resolution (5344×4008) images with its Scientific Camera (SciCam). Images collected with the SciCam cannot be streamed due to their large sizes, so this work uses NavCam imagery by default.

Two types of paths were designed for Astrobee. In the surveys of the Granite Lab, the paths were designed to emphasize translation. In the surveys of the ISS, the paths were designed to emphasize 360° rotation, approximately fixing the position of Astrobee. Figure 4 plots two paths against the corresponding model, one for a survey in the Granite Lab and one for a survey in the ISS. Throughout this work, different Astrobee units are referenced by their nicknames ("BSharp", "Queen", and "Bumble"), as used in the Granite Lab and ISS deployments.

3.2 Low-Movement Image Removal

Because Astrobee's flight speed is low compared to the NavCam sensor rate, the robot may capture sequential NavCam images which exhibit little to no movement between-frame. Data are processed by removing these
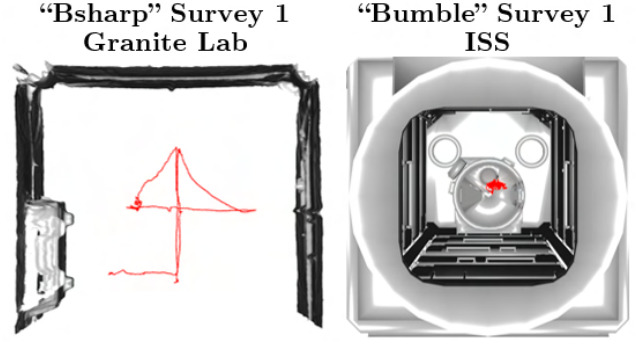
Figure 4: The paths of the Granite Lab surveys emphasize translational wall-scanning and the paths of the ISS surveys emphasize rotation.

low-movement images through feature matching and optical flow. Sparse features are detected and matched between two images using the pyramidal implementation of the Lucas-Kanade optical flow feature tracker [3]. If the distance between these sets of features is less than a maximum distance threshold, low movement is detected between the images and one of the images is discarded. Low-movement image removal reduces the number of image candidates for change detection from thousands to tens [21].

3.3 Camera Pose Estimation

FastCD requires an estimate of the pose of the camera in the World coordinate frame defined by the 3D model. The camera pose is estimated as a homogeneous transformation matrix, $\mathbf{T}^i_{4 \times 4}$, where i represents the index of the image-pose pair in the data sequence. The pose $\mathbf{T}^i$ can be computed from the body pose of the Astrobee robot in the World frame, $\mathbf{T}^i_{\text{Astrobee}}$, and the pose of the NavCam with respect to the Astrobee robot, $^{\text{Astrobee}}\mathbf{T}_{\text{NavCam}}$. The pose of the Astrobee robot is estimated at a rate of approximately 5 Hz with a factor graph-based localization system using an inertial measurement unit on the robot [10, 7, 33]. The extrinsic calibration of the frame of the NavCam with respect to the body frame of the Astrobee robot was performed using Kalibr [16, 12, 13, 19]. The pose is computed as

$$\mathbf{T}^i = ^{\text{World}}\mathbf{T}^i_{\text{Astrobee}} \, ^{\text{Astrobee}}\mathbf{T}_{\text{NavCam}}. \quad (1)$$

3.4 3D World Model Reconstruction

The NASA Ames Stereo Pipeline package generates a 3D map of the environment from Astrobee Navigation

Camera (NavCam), Hazard Camera (HazCam), and localization data [32, 23]. It registers image data from the NavCam with depth information from the HazCam using Theia structure-from-motion [34], and then fuses the depth point clouds into a mesh [2]. This method was used to reconstruct a 3D model of the Granite Lab environment included with the contributed dataset and used throughout this work.

3.5 Computing Inconsistencies

Given the camera projection matrix, $\mathbf{P}^i_{3\times4}$, the projection of an arbitrary 3D point in homogeneous coordinates, $\mathbf{X}^i_{4\times1}$, onto the plane of image i as a 2D pixel in homogeneous coordinates, $\mathbf{x}^i_{3\times1}$, is

$$\mathbf{x}^i = \mathbf{P}^i\mathbf{X}^i, \tag{2}$$

where the projection matrix

$$\mathbf{P}^i = \mathbf{K}\mathbf{R}^i\left[\mathbf{I}_{3\times3}\mid-\mathbf{t}^i\right] \tag{3}$$

is computed from the camera intrinsic calibration matrix $\mathbf{K}_{3\times3}$ and the rotation $\mathbf{R}^i_{3\times3}$ and translation $\mathbf{t}^i_{3\times1}$ components of $\left(\mathbf{T}^i\right)^{-1}$ that transform the World coordinates into NavCam coordinates. Inverting Eq. (2) gives

$$\mathbf{r}^i = \left(\mathbf{R}^i\right)^\intercal\mathbf{K}^{-1}\mathbf{x}^i, \tag{4}$$

the ray from the projection center of the camera through pixel $\mathbf{x}^i$ to the World. Each pixel of image $\mathcal{I}^i_{1280\times960}$ can then be back-projected onto the 3D model.

Inconsistencies are detected between a pair of images captured from poses $\mathbf{T}^i$ and $\mathbf{T}^{i+1}$ by creating a new intensity image, $\mathcal{I}^{i\rightarrow i+1}_{1280\times960}$, to store the content of $\mathcal{I}^i$ from pose $\mathbf{T}^{i+1}$ given the 3D model. The intersection, $\mathbf{X}^{i\rightarrow i+1}$, between a ray $\mathbf{r}^i$ from Eq. (4) and the 3D model is computed and and projected onto $\mathcal{I}^{i\rightarrow i+1}$ as

$$\mathbf{x}^{i\rightarrow i+1} = \mathbf{P}^{i+1}\mathbf{X}^{i\rightarrow i+1}, \tag{5}$$

where $\mathbf{P}^i$ is the camera projection matrix corresponding to pose $\mathbf{T}^i$. Since image $\mathcal{I}^i$ and image $\mathcal{I}^{i\rightarrow i+1}$ are at the same perspective, $\mathcal{I}^{i\rightarrow i+1}$ can be used to segment $\mathcal{I}^{i+1}$.

An accurate camera pose is unavailable for most robotic systems and the 3D model may be noisy. The pixel $\mathbf{x}^{i\rightarrow i+1}$ therefore has an associated uncertainty $\mathbf{\Sigma}_{3\times3}$ defined as

$$\mathbf{\Sigma} := \mathbf{\Sigma}_{\mathbf{x}^{i\rightarrow i+1}\mathbf{x}^{i\rightarrow i+1}}. \tag{6}$$

To account for this uncertainty, for each $\mathbf{x}^{i+1} \in \mathcal{I}^{i+1}$ compute the minimum Euclidean norm of the intensity difference to each pixel $\mathbf{z}$ of $\mathcal{I}^{i\rightarrow i+1}$ in a region, $\mathcal{R}$, around $\mathbf{x}^{i+1}$ as

$$d^{i\rightarrow i+1}(\mathbf{x}^{i+1}) = \min_{\mathbf{z}\in\mathcal{R}}\|\mathcal{I}^{i+1}(\mathbf{x}^{i+1}) - \mathcal{I}^{i\rightarrow i+1}(\mathbf{z})\|. \tag{7}$$

For a distance threshold $\tau^2 = 11.82$, the critical value of the χ^2_2 distribution corresponding to the 3σ bound on the standard normal distribution, the region $\mathcal{R}$ is defined as

$$\mathcal{R}(\mathbf{z}) = \begin{cases} \left(\mathbf{x}^{i+1} - \mathbf{z}\right)^\intercal\mathbf{\Sigma}^{-1}\left(\mathbf{x}^{i+1} - \mathbf{z}\right) < \tau^2 & \forall\mathbf{z}\in\mathcal{I}^{i\rightarrow i+1} \\ 0 & \text{otherwise} \end{cases}. \tag{8}$$

If no change is detected between the 3D model and images $\mathcal{I}^i$ and $\mathcal{I}^{i+1}$, each $\mathbf{x}^i \in \mathcal{I}^i$ should re-project onto $\mathcal{I}^{i+1}$, images $\mathcal{I}^{i\rightarrow i+1}$ and $\mathcal{I}^{i+1}$ should be identical, and $\forall\mathbf{x}^{i+1} \in \mathcal{I}^{i+1}$, $d^{i\rightarrow i+1}(\mathbf{x}^{i+1}) \approx 0$. If there is a mismatch between the 3D model and images due to a change in the environment, pixels corresponding to the change will re-project onto the wrong place in $\mathcal{I}^{i+1}$ and the change will be detected from a large $d^{i\rightarrow i+1}(\mathbf{x}^{i+1})$ value.

As originally highlighted in FastCD, this method has ambiguities when only two images are compared. A point $\mathbf{X}^i_c$ corresponding to a change in the 3D model generates two pixel locations on $\mathcal{I}^{i\rightarrow i+1}$. FastCD resolves this ambiguity using multiple pair-wise image comparisons [26]. Two pixels $\mathbf{x}^{i+1}$ and $\mathbf{x}^{i+2}$ belonging to the same change will project onto $\mathcal{I}^i$ at the same location. Therefore, pixels from different images which re-project onto the same region of $\mathcal{I}^i$ represent the real locations of change. Change regions are estimated in 2D by comparing each image with m neighboring images. The default value of $m = 4$ is used in this work unless otherwise noted. An example illustrating how inconsistencies are computed is included in Figure 5. After an inconsistency image with the 2D regions of change is computed, the inconsistency image is filtered through a sequence of image processing steps including erosion-dilation, contour filtering, and change region association.

3.6 Regions of Change

Once the regions containing changes are identified in 2D, the corresponding mean location of change, $\overline{\mathbf{X}}^i_c$, is estimated in 3D [26, 11]. Every region identified as a change has mean pixel location $\overline{\mathbf{x}}^i$ and covariance
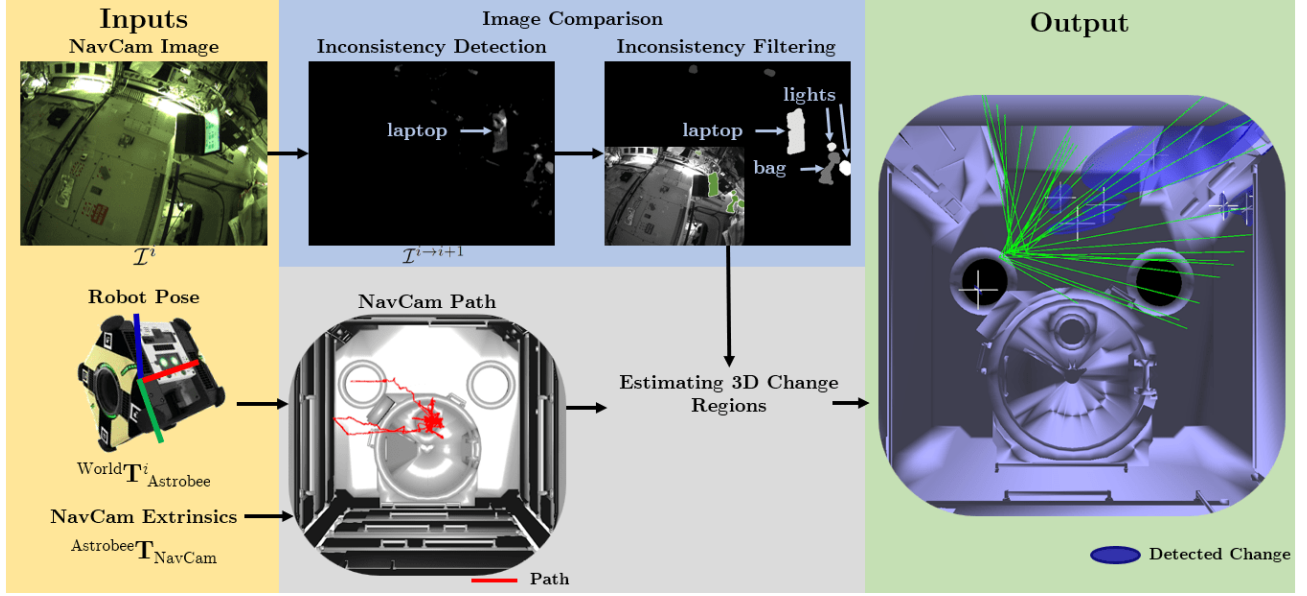
Figure 5: A laptop and lights are discovered in the ISS from a set of $n = 7$ Astrobee NavCam images. (Left) FastCD inputs include a sequence of $n = 7$ images, $\mathcal{I}^i$, and the pose at which each image was captured, $\mathbf{T}^i$. (Center Top) FastCD computes inconsistencies between images given the 3D model and filters the inconsistency image. (Center Bottom) Images are projected onto a 3D model given the path and pose of the camera to estimate 3D change regions. (Right) The 3D change regions are projected into the 3D model as change ellipses.

$\mathbf{\Sigma}^i$. The corresponding 3D point $\overline{\mathbf{X}}_c^i$ is computed in World coordinates by triangulating the mean location of change in each image. The 3D locations of change are estimated by solving

$$\mathbf{A}\overline{\mathbf{X}}_c = 0, \tag{9}$$

In Eq. (9),

$$\mathbf{A}_{3n \times 4} = \begin{bmatrix} \mathrm{S}\left(\overline{\mathbf{x}}^1\right)\mathbf{P}^1, & \ldots, & \mathrm{S}\left(\overline{\mathbf{x}}^n\right)\mathbf{P}^n \end{bmatrix}^\mathsf{T}, \tag{10}$$

n is the number of images, $\mathbf{P}^i$ is the projection matrix corresponding to pose $\mathbf{T}^i$, and $\mathrm{S}\left(\overline{\mathbf{x}}^i\right)$ is a skew-symmetric matrix generated from the elements of $\overline{\mathbf{x}}^i = \begin{bmatrix} x^i, y^i, w^i \end{bmatrix}^\mathsf{T}$ such that

$$\mathrm{S}(\overline{\mathbf{x}}^i) = \begin{bmatrix} 0 & -w^i & y^i \\ w^i & 0 & -x^i \\ -y^i & x^i & 0 \end{bmatrix}. \tag{11}$$

Eq. (9) is solved using singular value decomposition [26]. This procedure is repeated for the sigma points from every mean and covariance of the same region in every image to efficiently estimate 3D change regions without requiring a dense reconstruction of the environment. The mean of the points $\overline{\mathbf{X}}_c$ represent the regions in 3D where changes occurred. The covariances associated with $\overline{\mathbf{X}}_c$ are used to draw covariance ellipses to characterize the uncertainty of the estimate.

4.   Results and Limitations

FastCD was applied to data collected by Astrobee units in two environments. The "Bsharp" Astrobee unit surveyed the controlled Granite Lab environment (Section 4.1) and the "Bumble" and "Queen" Astrobee units surveyed the uncontrolled ISS environment (Section 4.2). In this qualitative study, FastCD detects large objects appearing in multiple sequential image frames.

4.1   Granite Lab Results

The "Bsharp" Astrobee unit surveyed the Granite Lab five times. The robot followed mostly translational paths, scanning each wall of the Granite Lab. The first four surveys (`20230419_bsharp_survey1`, `20230419_bsharp_survey2`, `20230419_bsharp_survey3`, and `20230419_bsharp_survey4`) are included in the released dataset. For each of these surveys, one or more confounding objects realistic to the ISS environment are introduced, including a cargo bag, a crate, a cable, and a static Astrobee unit, shown in Figure 6. The fifth survey is not included in the released dataset. This survey was used to build the 3D model (`model.obj`) included with each Granite Lab survey, following the framework described in Section 3.4. Object discovery in Granite Lab data is shown in Figure 7.
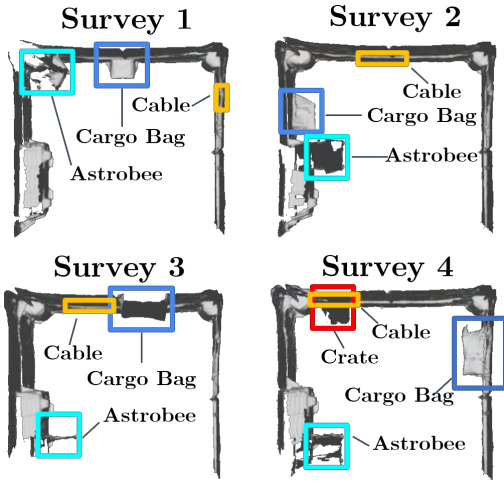
Figure 6: Objects are introduced at known locations to the Granite Lab environment for each of four surveys.
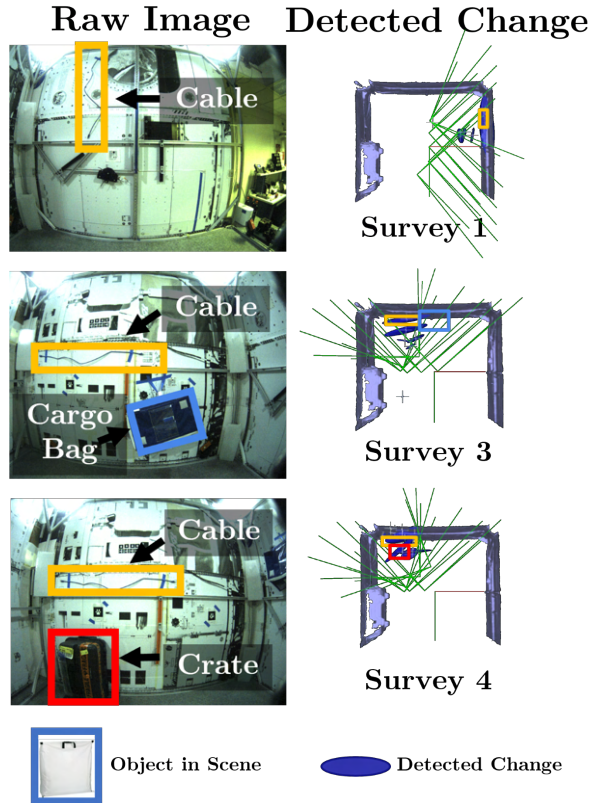
Figure 7: FastCD detects several objects added to the Granite Lab environment. (Top Row) Cable detection from Survey 1. (Middle Row) Cable and cargo bag detection from Survey 3. (Bottom Row) Cable and crate detection from Survey 4.

For each survey, the maximum number of between-image comparisons, m, was varied from 2 to 6, using the FastCD default $m = 4$ as a center point. Results were consistent with those found in [26]: with $m < 4$, the results were much noisier with increased false-positive detections. However, with each Granite Lab survey, the quality of the results did not increase with $m \geq 4$. False positives were occasionally found near the camera position, although less frequently, with $m \geq 4$. While the robot itself is technically a changing object between images in the scene, this result could be mitigated by filtering results within a certain proximity to the camera. After removing artifacts, these early FastCD results appear promising.

To analyze the accuracy of the results, the 3D coordinates of the known changes and of the detected changes are necessary ground truth data. The latter data are difficult to evaluate directly. Furthermore, the output model provides an overview of the area where changes are detected as a cluster of ellipses, and therefore a single item may be highlighted as several changes. However, for all surveys, the majority of ellipses are clustered around the location of the added object.

Compute-constrained processors, including the 4-core, 2.5 [GHz] Snapdragon 805-based IFC6501s used by the Astrobee units, must meet real-time rates for autonomy software while running multiple processes concurrently [31]. Runtime must be verified to show, at the very least, standalone real-time performance of any autonomy algorithm. Computational timing data were collected for the first Granite Lab survey data on a computer with a 4-core, 3.20 [GHz] Intel i5-4570 CPU. Table 1 reports runtimes for three FastCD computational processes based on the number of images, n, in the batch used to compute changes. All other FastCD parameters were kept at defaults. Data loading accounts for most of the runtime. Data are loaded as `.png` images and `.xml` poses and camera intrinsic parameters. For online change detection, these data could be retrieved from queues and made available to other processes such as localization.

Table 1: FastCD Runtime [s]

n	Data Loading	Inconsistencies	3D Change
2	0.844	0.311	0.054
3	0.800	0.606	0.076
4	0.843	1.156	0.134
5	0.835	1.891	0.161
6	0.842	2.280	0.266
Per Image	−	≈ 0.281	−

The most computationally intensive component of FastCD is computing inconsistencies, with computation time scaling with n. For $n < 7$, FastCD localizes scene changes in < 2.5s, fast enough to present actionable information about the environment to a mobile robot. However, for a change detection algorithm running on Astrobee, a $10\times$ slowdown is expected due to the computational load of other processes. Real-time performance of FastCD on Astrobee units is expected to require further optimization.

4.2 ISS Activity Results

The "Bumble" and "Queen" Astrobee units each performed three surveys of the ISS. The robots followed mostly rotational paths to scan the ISS Japanese Experiment Module (JEM) autonomously. All six of these surveys are included inh released dataset (20220608_bumble_survey1, 20220608_bumble_survey2, 20220608_bumble_survey3, 20220608_queen_survey1, 20220608_queen_survey2, and 20220608_queen_survey3). Unlike in the Granite Lab, the environment was not modified for the purpose of this work: the ISS was surveyed as-is. The complexity of the ISS environment precludes reconstruction of a clean map from robot survey data, so this work uses the Astrobee simulation JEM model [22]. Samples of FastCD object discovery on the ISS data are shown in Figure 8.

The results in Figure 8 highlight the importance of noise filtering steps on the computed 2D inconsistencies. The ISS, like many human habitats, is cluttered. This makes it more difficult to detect changes at both local and full-scene scales. The FastCD algorithm uses image processing functions to remove insignificant regions estimated as change regions. Increasing the number of geometric projection operations by increasing m also reduces noise. Future work using semantic labeling could further reduce noise prior to 3D location of change estimation. It is also important to note that the 3D Model used for the ISS Activities is a simulated model of the JEM and was not generated from image and pose data in the same way the Granite Lab 3D model was generated. Since the model was not reconstructed from real data, it introduces additional uncertainties due to the Sim2Real gap.
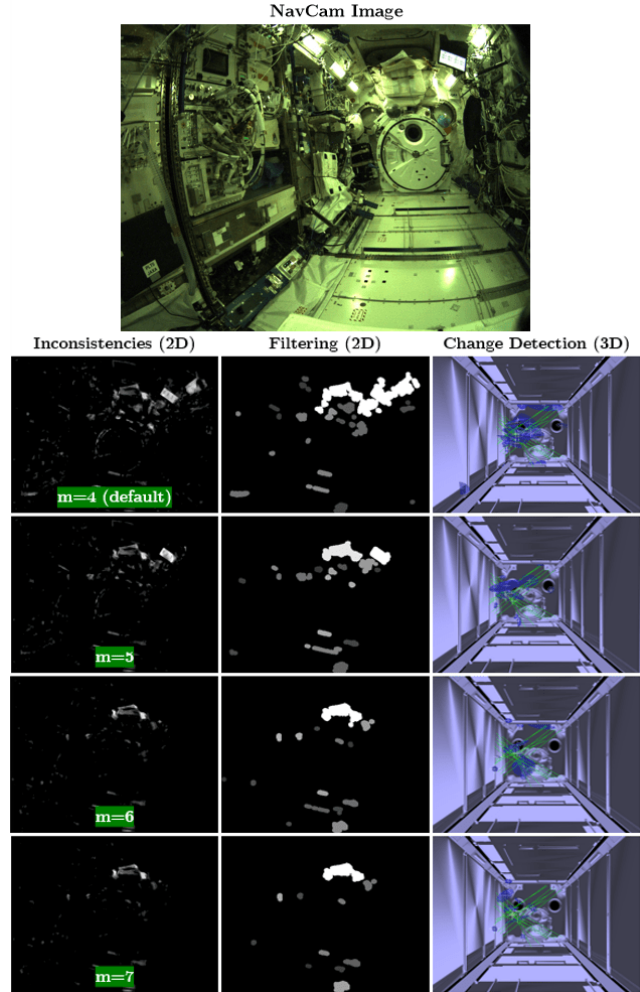
Figure 8: As the maximum number of between-image comparisons, m, increases, the number of detected changes decreases, removing spurious detections.

4.3 Limitations

FastCD assumes a good estimate of the camera pose. One way this limitation can be addressed for Astrobee is to perform bundle adjustment against a known 3D map of the environment to obtain an accurate ground truth localization. For offline applications, this method of obtaining the ground truth pose may be acceptable. However, for online surveying and change detection, Astrobee would not have access to ground truth pose information and an inaccurate camera pose could lead to false positive or undetected changes.

Astrobee localization accuracy suffers most when the movement between consecutive images is low and

when the number of features shared between consecutive images is low. This is usually due to rotation-only movement with little or no translation, introducing cheirality issues [33]. As illustrated in Fig. 4, the Granite Lab surveys used T-shaped paths designed to increase translation in the survey. The ISS Activity surveys used rotate-in-place paths. These rotation-only surveys suffer the most from localization inaccuracy, increasing change detection errors. Other hardware- and environment-specific requirements include good lighting, prominent visual features, and a sufficient field of view for features to overlap between frames.

5. Discussion and Future Work

Motivated by the promise of assistive robotic technology in space environments, this work applied FastCD, a fast scene change detection algorithm, to object discovery for habitat maintenance. After discussing the specific considerations for scene change detection on a resource-constrained mobile robot in a space environment, FastCD was demonstrated on data collected with Astrobee units in the Granite Lab and aboard the ISS. The dataset used in this work is publicly released at https://bit.ly/astrobee_fastcd_data.

Future work could perform geometric scene change detection in real-time with multiple cameras, use images with semantic annotations to improve change localization, or feedback detected changes in real-time into an anomaly recovery architecture [20]. Future work could also improve on elements presented here. Currently, the limitations of FastCD for anomaly detection include sensitivity to object size, sensitivity to camera pose accuracy, and sensitivity to camera resolution. Robustness could also be further improved through detection pruning.

Future work may also include runtime analysis on embedded processors. This work found computing inconsistencies is the most resource-consuming process and is related to n, the number of images in a batch. Computation time should also depend on m, the maximum number of comparisons used for detection in the FastCD algorithm. Investigating the impact of m on timing will be important to keep computational expense within the resource budget of free-flyers such as Astrobee. Interesting extensions which may improve the accuracy of the FastCD algorithm include change detection labeling and semantics as well as robustness to illumination change.

Finally, this type of work unlocks numerous benefits

for assistive robotics. Because FastCD estimates locations of change with uncertainty, when an environment map needs updating, only the local changes need to be identified and remapped. This feature is applicable to any robotic system, whether in space or terrestrial. Moreover, methods like FastCD allow Astrobee or other free-flyers to feasibly work alongside astronauts in space habitats, thus reducing astronaut workload. Furthermore, a great benefit of FastCD is its speed, which is especially useful for a resource-constrained mobile platform. As the world increasingly turns to automation to meet its needs, it is more important than ever to enable robots with the capability of responding to dynamic, human environments.

Acknowledgements

References

[1] Pablo F. Alcantarilla, Simon Stent, German Ros, Roberto Arroyo, and Riccardo Gherardi. Street-View Change Detection with Deconvolutional Networks. *Auton. Robot.*, 42: 1301–1322, 2018.

[2] Ross A. Beyer, Oleg Alexandrov, and Scott McMichael. The Ames Stereo Pipeline: NASA's Open Source Software for Deriving and Processing Terrain Data. *Earth Space Sci.*, 5, 2018.

[3] Jean-Yves Bouguet et al. Pyramidal Implementation of the Affine Lucas-Kanade Feature Tracker: Description of the Algorithm. *Intel Corporation*, 5(1-10):4, 2001.

[4] Maria G. Bualat, Trey Smith, Ernest E. Smith, Terrence Fong, and Dw Wheeler. Astrobee: A New Tool for ISS Operations. In *2018 SpaceOps Conference*, Marseille, France, May 2018. American Institute of Aeronautics and Astronautics. ISBN 978-1-62410-562-3. doi: 10.2514/6.2018-2517.

[5] Maria G. Bualat, Jonathan S. Barlow, Jose V. Benavides, Brian Coltin, Lorenzo Flückiger, Marina Gouveia Moreira, Kathryn M. Hamilton, Aric Katterhagen, Ryan Soussan, Trey Smith, and The Astrobee Team. Astrobee On-Orbit Commissioning. *IAF Int. Conf. Space Ops.*, 16(SpaceOps-2021,9,x1518):1–9, 2021.

[6] Roberto Carlino, Jonathan Barlow, Jose Benavides, Maria Bualat, Aric Katterhagen, Yunji Kim, Ruben Garcia Ruiz, Trey Smith, and Andres Mora Vargas. Astrobee Free Flyers: Integrated and Tested. Ready for Launch! *IAF Int. Astr. Cong.*, 70:1–9, 2019.

[7] Luca Carlone, Zsolt Kira, Chris Beall, Vadim Indelman, and Frank Dellaert. Eliminating Conditionally Independent Sets in Factor Graphs: A Unifying Perspective Based on Smart Factors. In *IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 4290–4297, 2014.

[8] Turgay Celik. Unsupervised Change Detection in Satellite Images Using Principal Component Analysis and K-Means Clustering. *IEEE Geosci. Remote Sens. Lett.*, 6(4):772–776, 2009.

[9] Jason C. Crusan, R. Marshall Smith, Douglas A. Craig, Jose M. Caram, John Guidi, Michele Gates, Jonathan M. Krezel, and Nicole B. Herrmann. Deep space gateway concept: Extending human presence into cislunar space. *IEEE Aerospace Conference Proceedings*, 2018-March:1–10, 2018. ISSN 1095323X. doi: 10.1109/AERO.2018.8396541.

[10] Frank Dellaert. Factor Graphs and GTSAM: A Hands-On Introduction. *Georgia Institute of Technology, Tech. Rep*, 2:4, 2012.

[11] Wolfgang Förstner and Bernhard P Wrobel. *Photogrammetric Computer Vision*. Springer, 2016.

[12] Paul Furgale, Timothy D. Barfoot, and Gabe Sibley. Continuous-Time Batch Estimation Using Temporal Basis Functions. In *IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 2088–2095, 2012.

[13] Paul Furgale, Joern Rehder, and Roland Siegwart. Unified Temporal and Spatial Calibration for Multi-Sensor Systems. In *IEEE/RSJ Int. Conf. Intell. Robot. Sys. (IROS)*, pages 1280–1286, 2013.

[14] Mani Golparvar-Fard, Feniosky Peña-Mora, and Silvio Savarese. Monitoring Changes of 3D Building Elements From Unordered Photo Collections. In *IEEE Int. Conf. Comput. Vis. (ICCV)*, pages 249–256, 2011.

[15] Tanaka Kanji. Detection-By-Localization: Maintenance-Free Change Object Detector. In *IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 4348–4355, 2019.

[16] ETH Autonomous Systems Laboratory. Kalibr. [Online] Available: https://github.com/ethz-asl/kalibr, 2023.

[17] Emma Lehnhardt, Sean Fuller, Jarrett Quasny, Dylan Connell, Christina Zaid, and Kate Halloran. The Gateway as a Building Block for Space Exploration and Development. In *IAF Int. Astr. Cong.*, volume 73, 2022.

[18] Yan Li and Lei Li. A Novel Split and Merge EM Algorithm for Gaussian Mixture Model. In *Int. Conf. Nat. Comput.*, volume 6, pages 479–483. IEEE, 2009.

[19] Jérôme Maye, Paul Furgale, and Roland Siegwart. Self-Supervised Calibration for Robotic Systems. In *IEEE Intell. Veh. Symp. (IV)*, pages 473–480, 2013.

[20] Ian D. Miller, Ryan Soussan, Brian Coltin, Trey Smith, and Vijay Kumar. Robust Semantic Mapping and Localization on a Free-Flying Robot in Microgravity. In *IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 4121–4127, 2022.

[21] NASA. Astrobee Robot Software. [Online] Available: https://github.com/nasa/astrobee, 2023.

[22] NASA. Astrobee Media. [Online] Available: https://github.com/nasa/astrobee_media.git, 2023.

[23] NASA. ISAAC (Integrated System for Autonomous and Adaptive Caretaking). [Online] Available: https://github.com/nasa/isaac, 2023.

[24] A.S. Otero, A. Chen, D.W. Miller, and M. Hilstad. SPHERES: Development of an ISS Laboratory for formation flight and docking research. In *Proceedings, IEEE Aerospace Conference*, volume 1, pages 1–59–1–73, Big Sky, MT, USA, 2002. IEEE. ISBN 978-0-7803-7231-3. doi: 10.1109/AERO.2002.1036828.

[25] Emanuele Palazzolo and Cyrill Stachniss. Change Detection in 3D Models Based on Camera Images. In *Workshop on Planning, Perception and Navigation for Intelligent Vehicles at the IEEE/RSJ Int. Conf. on Intell. Robot. Sys. (IROS)*, 2017.

[26] Emanuele Palazzolo and Cyrill Stachniss. Fast Image-Based Geometric Change Detection Given a 3D Model. In *IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 6308–6315, 2018.

[27] Rongjun Qin, Jiaojiao Tian, and Peter Reinartz. 3D Change Detection–Approaches and Applications. *ISPRS J. Photogramm*, 122:41–56, 2016.

[28] Richard J. Radke, Srinivas Andra, Omar Al-Kofahi, and Badrinath Roysam. Image Change Detection Algorithms: A Systematic Survey. *IEEE Trans. Image Process.*, 14(3): 294–307, 2005.

[29] Yossi Rubner, Carlo Tomasi, and Leonidas J Guibas. The Earth Mover's Distance as a Metric for Image Retrieval. *Int. J. Comput. Vis.*, 40(2):99, 2000.

[30] Jamie Santos, Holly Dinkel, Julia Di, Paulo V.K. Borges, Marina Moreira, Brian Coltin, and Trey Smith. Unsupervised Change Detection for Space Habitats Using 3D Point Clouds. In *AIAA SciTech F.*, 2024.

[31] Trey Smith, Jonathan Barlow, Maria Bualat, Terrence Fong, Christopher Provencher, Hugo Sanchez, and Ernest Smith. Astrobee: A New Platform for Free-Flying Robotics on the International Space Station. In *Int. Sympos. Artif. Intell. Robot. Autom. in Space (i-SAIRAS)*, 2016.

[32] Trey Smith, Maria Bualat, Abiola Akanni, Oleg Alexandrov, Laura Barron, J Benton, Brian Coltin, Terry Fong, Janette Garcia, Kathryn Hamilton, Lewis Hill, Marina Moreira, Robert Morris, Nicole Ortega, Joseph Pea, Jonathan Rogers, Misha Savchenko, Khaled Sharif, and Ryan Soussan. ISAAC: An Integrated System for Autonomous and Adaptive Caretaking. In *ISS R&D Conference*, 2021.

[33] Ryan Soussan, Varsha Kumar, Brian Coltin, and Trey Smith. AstroLoc: An Efficient and Robust Localizer for a Free-flying Robot. In *IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 4106–4112, 2022.

[34] Chris Sweeney. Theia Multiview Geometry Library: Tutorial & Reference. [Online] Available: http://theia-sfm.org, 2023.

[35] Aparna Taneja, Luca Ballan, and Marc Pollefeys. Image Based Detection of Geometric Changes in Urban Environments. In *IEEE Int. Conf. Comput. Vis. (ICCV)*, pages 2336–2343. IEEE, 2011.

[36] Aparna Taneja, Luca Ballan, and Marc Pollefeys. City-Scale Change Detection in Cadastral 3D Models Using Images. In *IEEE/CVF Int. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pages 113–120, 2013.

Authors

Holly Dinkel 🔗 🏠 is pursuing a Ph.D. in aerospace engineering at the University of Illinois Urbana-Champaign where she researches robotic caretaking as a NASA Space Technology Graduate Research Fellow with the NASA Ames Research Center Intelligent Robotics Group and the NASA Johnson Space Center Dexterous Robotics Laboratory.

Julia Di 🔗 🏠 is pursuing a Ph.D. in mechanical engineering at Stanford University where she researches tactile sensing and perception. She was a NASA Space Technology Graduate Research Fellow with the NASA Ames Research Center Intelligent Robotics Group and NASA Jet Propulsion Laboratory Mobility and Robotic Systems Section.

Keenan Albee 🔗 is a Robotics Technologist in the Maritime and Multi-Agent Autonomy group at the Jet Propulsion Laboratory, California Institute of Technology. He received a Ph.D. in Aeronautics and Astronautics from the Massachusetts Institute of Technology. He is interested in motion planning under uncertainty, information-aware planning, and real-time model learning.

Jamie Santos 🔗 completed a M.S. in Complex and Adaptive Systems at Chalmers University in Gothenburg, Sweden. She researches change detection to enable NASA's free-flying Astrobee robots to detect anomalies on the International Space Station with the NASA Ames Research Center Intelligent Robotics Group.

Paulo Borges 🔗 🏠 is a Principal Research Scientist in the Robotics and Autonomous Systems Group at CSIRO in Brisbane, Australia. He completed a Ph.D. in Electronic Engineering from Queen Mary University of London, London, United Kingdom. His research interests include robotic automation for the manufacturing, energy, and agriculture industries, bridging industry, innovation, and research.

Marina Moreira 🔗 is a research engineer in the NASA Ames Research Center Intelligent Robotics Group. She completed a M.S. in aerospace engineering from Instituto Superior Técnico, Lisbon, Portugal. She develops and maintains open-source software for the ISAAC problem and is interested in problems related to robotic systems and control.

Oleg Alexandrov 🔗 is a Research Scientist in the NASA Ames Research Center Intelligent Robotics Group. He received a Ph.D. in applied mathematics from the University of Minnesota. His research focuses on mapping using satellite and robot images.
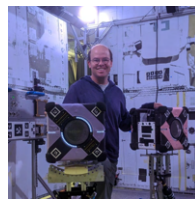
Brian Coltin 🔗 🏠 is a Computer Scientist in the NASA Ames Research Center Intelligent Robotics Group where he works on the Astrobee robot, the VIPER rover, and flood mapping. He earned a Ph.D. in Robotics from Carnegie Mellon University. His research interests include planning, scheduling, multi-robot coordination, localization, and computer vision.

Trey Smith 🔗 🏠 is a Computer Scientist in the NASA Ames Research Center Intelligent Robotics Group where he works on the Astrobee robot and the VIPER rover. He earned a Ph.D. in Robotics from Carnegie Mellon University. His research interests include planning, scheduling, multi-robot coordination, mapping, and computer vision.