Robust Semantic Mapping and Localization on a Free-Flying Robot in Microgravity

Ian D. Miller[1], Ryan Soussan[2], Brian Coltin[2], Trey Smith[2], and Vijay Kumar[1]

***Abstract*— We propose a system that uses semantic object detections to localize a microgravity free-flyer. Many applications require absolute localization in a known reference frame, such as the execution of waypoint trajectories defined by human operators. Classical geometric methods build a map of point features, which may not be able to be associated after lighting or environmental changes. By contrast, semantics remain invariant to changes up to the robustness of the detection algorithm and motion of the semantic objects. In this work, we describe our approaches for both offline semantic map generation as well as online localization against a semantic map, intended to run in real-time on the robot. We additionally demonstrate how our semantic localizer outperforms image-feature matching in some cases, and show the robustness of the algorithm to environmental changes. Crucially, we show in our experiments that when semantics are used to supplement point features, localization is always improved. To our knowledge, these experiments demonstrate the first use of learned semantics for localization on a free-flying robot in microgravity.**

I. Introduction

The Astrobee free-flying robots [1] have been operating inside the International Space Station (ISS) since 2019. They can host free-flying robot research, act as mobile cameras for ground controllers, and collect sensor surveys [2]. Operators use a 3D graphical interface to plan motion waypoints by positioning them inside a CAD model of the ISS. Executing these motions requires Astrobee's onboard localization software to provide position estimates in the absolute ISS coordinate frame used by the CAD model. Robot localization is an extremely well-studied problem [3], [4], [5], but the single monocular imager, limited computational capability, cluttered, truly 3D environment, and necessary robustness of intra-vehicular robotics (IVR) on the ISS give rise to unique challenges.

The strategy currently employed by Astrobee [6], [7] involves building an *a priori* sparse map of BRISK point features [8] located in the ISS coordinate frame. This map is constructed using offline structure-from-motion (SfM) algorithms. For online localization, features are detected in an image, and a lookup is performed to the sparse map database to determine the robot's pose. While accurate when successful, we argue that reliance on geometric features such as BRISK leads to a brittle algorithm. Feature matching fails in the presence of lighting or environmental changes that make a location that is still recognizable by a human unrecognizable for the robot. Environmental changes on the ISS can include reconfiguration of experiments or stowage containers being attached to the deck. Lighting changes can be mitigated by adjusting exposure time and selecting features intelligently [9], but changes such as individual light sources moving or turning on and off changes the structure and direction of shadows, which cannot be compensated for with whole-image adjustments.

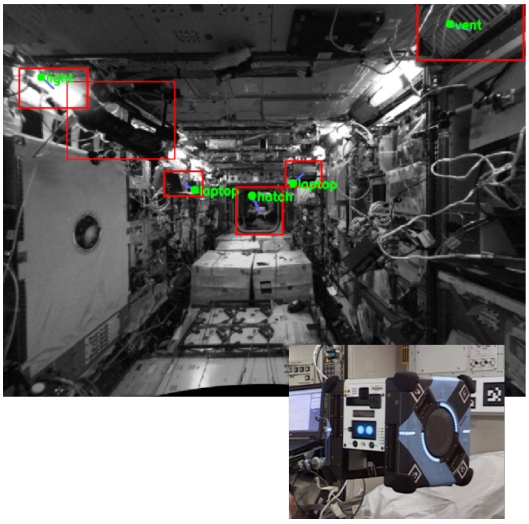

Fig. 1: *Top*: Object detections on the ISS, shown in red, with map objects projected into the image in green. Blue lines indicate the associations between detections and map objects. *Bottom*: Astrobee robot.

In this work we argue that the *semantics* of the environment provide robust cues that the robots can use to localize. In theory, the identity of an object is completely invariant to viewpoint, lighting, and configuration, while features relying directly on pixel values are not. In practice, the invariance of semantics depends on the detector used. In recent years, Deep Convolutional Neural Networks (DCNNs) such as EfficientDet [10] have been shown to achieve strong robustness as well as computational efficiency, even in constrained environments. Such detectors show great promise for detecting key objects to act as robust localization landmarks. In addition, the resulting object maps are human-interpretable, enabling other high-level tasks such as navigation to objects or directed exploration, whereas a sparse feature point is much less informative.

While we motivate and test our method on Astrobee datasets from the ISS, we note that our localizer can gen-

We gratefully acknowledge the support of NVIDIA, ONR Grant N00014-20-1-2822, and NSF Grant CCR-2112665. Ian Miller acknowledges support from a NASA Space Technology Research Fellowship.

[1] Ian D. Miller and Vijay Kumar are with the GRASP Lab, University of Pennsylvania, Philadelphia, PA 19104. Corresponding author: iandm@seas.upenn.edu

[2] Trey Smith, Ryan Soussan, and Brian Coltin are with the NASA Ames Intelligent Robotics Group, Moffett Field, CA 94035

eralize to a wide variety of applications. For instance, semantics additionally offer invariance to seasonal, viewpoint, and sensing changes, rendering them attractive for many terrestrial applications [11]. To our knowledge, this work constitutes the first application of semantic object detection to localization of free-flying robots in microgravity, and the unique robustness, environmental, and computational challenges therein.

Our contributions in this work are as follows:

- We present a novel method for semantic object-centric map construction given image poses and object detections, bypassing the data-association problem.
- We develop a complete system for global localization using the object-centric semantic map, capable of operating on computationally constrained hardware.
- We evaluate our method on a variety of datasets from the ISS, showing superior robustness compared to the existing feature-based system.

II. Related Work

A. Geometric Localization

Early localization approaches built a database of sparse image features to localize against. In [4], the authors match SIFT features between stereo cameras. They initialize the feature locations based on stereo depth and relocalize against previously seen features in the map. ORB-SLAM2 [12] takes a similar approach, building an ORB feature map and using a bag-of-words approach for relocalization. In addition, the current Astrobee localization system [6], [7] uses a sparse feature map composed of BRISK features. While these methods are effective at relocalization not long after the map was built, or performing online simultaneous localization and mapping (SLAM), they are only as robust as the chosen feature detector and descriptor across lighting or environment changes.

B. Semantic Localization

The earliest forms of object-based localization detected unique landmarks of known position, incorporating those measurements with odometry in a Bayesian framework. Lenser *et al*. [13] use colored posts located around a field coupled with a modified Monte Carlo particle filter to localize quadrupedal robots on a field for RoboCup. However, the method assumes that all landmarks are unique in appearance, making the problem of associating detections with map objects trivial.

In a work most similar to ours, Bavle *et al*. [5] localize a quadrotor in by fusing ground plane detections, stereo visual-inertial odometry (VIO), and semantic object detections in a particle filter, but treat the world as primarily 2D with a clear ground plane, an assumption that breaks on the ISS. Bowman *et al*. [14] propose a method for probabilistically associating map objects and detections, though this comes at the cost of somewhat more expensive optimization. Taking a different apprach, Ananti *et al*. [15] localize a robot in a two-dimensional object map using a particle filter and heatmaps of object-detection likelihood in image space, but this again makes a 2D assumption. In [16], the authors are able to relocalize in a known map viewed from a different angle by building semantic maps of cuboids from separate datasets, and registering these two maps to each other. Relocalization, however, requires aligning two maps, and the method cannot register a single image to a map.

Recently, dense semantic methods have grown in popularity. Instead of simply representing the map as a set of object classes and locations, Liu *et al*. [17] use a dense point-cloud representation, where each point is given a class. Objects are first matched by clustering classes and building descriptors on a semantic graph, followed by dense alignment of the objects' geometries. Gawel *et al*. [11] localize a semantically segmented image in a dense top-down map by building a graph descriptor based on semantic topologies in both spaces. While effective, dense methods require significantly more computation to handle the complex maps and matching. In addition, they require training semantic segmentation models, a process much more time-consuming than bounding-box labeling for object detection.

C. Semantic Map Building

Many methods seek to build high quality semantic maps in an online fashion. SLAM++ [18] uses a depth camera to detect object poses and build an object-level graph of objects of known geometry to describe the map. Kimera [19] performs semantic SLAM, utilizing VIO and building mesh representations of each semantic object, thereby not requiring known *a priori* object models. These methods also use depth sensors or stereo to aid in initializing object locations. By contrast, EAO-SLAM [20] requires only a monocular imager, and exploits the motion of the camera to determine the position and size of objects. However, in order to associate objects between frames, the method builds on ORB-SLAM2 [12] to match features contained within the object bounding boxes. By contrast, we do not build a map online, both because it is known *a priori* and to limit computation.

D. Localization for Space Robotics

Much of the literature for robot localization in space revolves around the non-cooperative rendezvous problem [21], where a robot attempts to rendezvous with some object in space using visual information. This problem differs significantly from ours in that there is essentially only one goal object floating in space, as opposed to the cluttered environment faced by IVR. The SPHERES free-flyer uses fixed infrared beacons in the ISS to localize [22], and Astrobee currently uses image features [7]. JAXA's Int-Ball 1 free-flyer localizes relative to a known visual fiducial [23]. To our knowledge, this work is the first application of learned semantics to the IVR localization problem in microgravity.

III. Method

In order to use semantic objects for localization, we must first detect these objects, as well as obtain a semantic object map to localize against. We first discuss these two problems, and finally address their integration in the localization algorithm, as well as the overall system design.

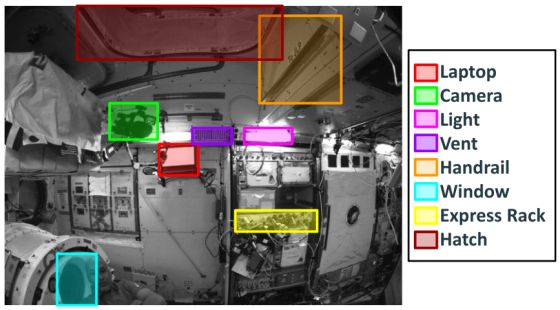

Fig. 2: Astrobee image with labelled instances of each class.

A. Object Detection

The Astrobee's monocular greyscale camera has significant fisheye distortion, so we rectify the image during pre-processing [24]. This choice allows us to use standard projective geometry methods for the rest of the algorithm.

We use the EfficientDet [10] model, in its smallest `lite0` variety. Training and validation is performed with a 90-10 split on 2351 bounding boxes, all taken from 132 images from the execution of a single Astrobee trajectory. We additionally use RandAug [25] for data augmentation in the context of TensorFlow Lite Model Maker. Our labels span across 8 classes, examples of which are shown in Fig. 2.

B. Map Building

1) Offline registration: We construct semantic maps offline to localize against. Given per-image detections from the trained detector, we also require high quality image poses. To do this, we first select a subset of the images from the dataset, avoiding overly redundant images while ensuring sufficient overlap. We then run an SfM pipeline [7] to create a bundle-adjusted pose graph of these images. These images are then registered against an existing sparse map of the ISS and bundle adjusted again. Finally, we run the existing sparse map localizer [6] on the entire dataset, using the newly-constructed map. Because the features in the new map are from images collected during the same ISS activity, environmental changes are minimal and registration is of very high quality. Once this procedure is complete, we have associated pose and object detections.

2) Notation: Let there be C object classes and K images in total. Let $c \in \{1, \dots, C\}$ be a class index. Let $D_k^c = \{d_{k,n}^c\}_{n=1,\dots,|D_k^c|}$ be the set of object detections for class c in image k, and p_k be the camera pose of image k. Then $\mathcal{D}^c = \{(D_k^c, p_k)\}_{k=1,\dots,K}$ is the set of object-detection-set/pose pairs for object class c. For simplicity and clarity of notation, we sometimes suppress the c indices.

3) Heatmap Generation: Next, we find candidate 3D object positions by searching for sufficiently large local maxima in object detection heat maps. The heat map construction process is described in Algorithm 1. We voxelize the map volume into L voxels, where each voxel l has position b_l, and for each class c define an object detection heat map $H^c = \{h_l^c\}_{l=1,\dots,L}$. The heat map value h_l^c should be large if the robot detects an object of class c when it looks toward b_l from diverse viewpoints.

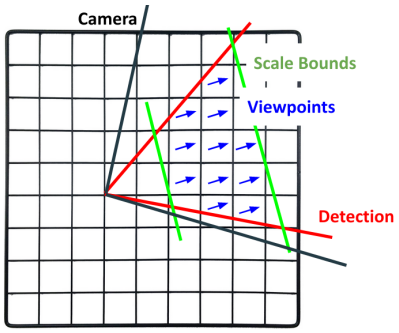

Fig. 3: Generation of the per-class object heatmaps.

Every detection $d_{k,n}$ with pose p_k defines a frustum, shown in red in Fig. 3. We clip this frustum by constraining the scale of the object based on knowledge of the approximate sizes of objects. These bounds are shown in green in Fig. 3, and we denote the resulting clipped frustum $F^c(d_{k,n}^c, p_k)$. Let $R(p)$ and $X(p)$ be the rotation matrix and position vector, respectively, of pose p. We then compute the object viewpoint for each cell in the detection frustum Dir, shown in blue in Fig. 3. We want to upweight cells which are contained in detection frusta from a variety of viewpoints, and Algorithm 1 uses the efficient heuristic of adding to h_l when the viewpoint changes significantly. The resulting per-class 2D heatmap projections of these weights are shown in Fig. 4.

4) Final Map Refinement: We then detect sufficiently large local maxima, and use these as our initial object locations. We let the final map be $\mathcal{M} = \{M^c\}_{c=1,\dots,C}$, where $M^c = \{m_1^c, \dots, m_{|M^c|}^c\}$ is the set of locations of $|M^c|$ objects of class c. Note that throughout this entire process, we avoid any explicit data association.

Algorithm 1 Per-class Heatmap Building

Input: Pose sequence p_k, detections d_k, constant δ
Output: Heatmap h_l

$V_l \leftarrow [0,0,0]^T \ \forall \ l \in \{1, \dots, L\}$
$h_l \leftarrow 0 \ \forall \ l \in \{1, \dots, L\}$
for $k \in \{1, \dots, K\}$ **do**
  $Dir \leftarrow R(p_k)[0,0,1]^T$
  for $n \in \{1, \dots, |D_k|\}$ **do**
    for $l \in \{1, \dots, L \mid b_l \in F(d_{k,n}, p_k)\}$ **do**
      $\Delta\theta \leftarrow \arccos(Dir \cdot V_l)$
      if $\Delta\theta > \delta$ **then**
        $V_l \leftarrow Dir$
        $h_l \leftarrow h_l + \Delta\theta$
      end if
    end for
  end for
end for

While the resulting maps are reasonable, we can nonetheless perform further refinement. We project the estimated object positions into each image using its known pose p_k. We greedily assign detections to objects on a per-class basis,

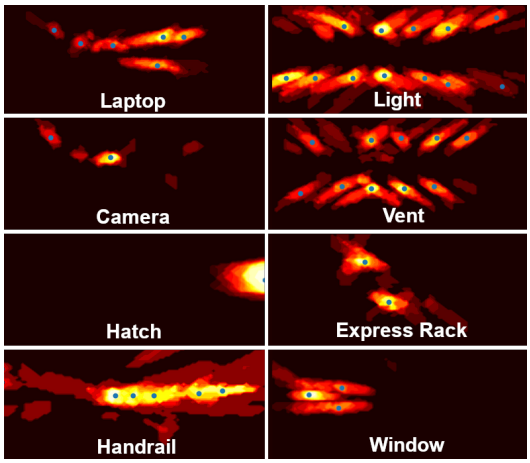

Fig. 4: Per-class 2D heatmap projections built from single dataset. Detected maxima are shown by blue dots.

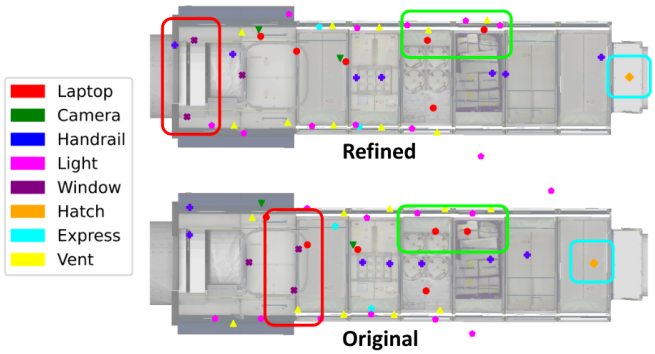

Fig. 5: Semantic maps overlaid onto the Japanese Experiment Module (JEM). The lower map shows the initial map after heatmap maximum detection, the upper map after pose graph optimization. The boxes highlight changes between the maps.

and thereby build a factor graph [26]. In this graph, the object positions and poses p_k act as nodes to be optimized, while projection factors connect the two, and prior factors anchor the robot poses to the initial estimates. We optimize this graph using GTSAM [26], and show the resulting map in Fig. 5. We find the refined map to be of higher qualitative quality than the initial map, as well as ultimately yielding quantitatively more accurate localization.

C. Localization

The Astrobee localization system uses a graph optimization formulation, which we summarize here but is described in more detail in [6]. The system maintains a pose graph over a sliding window of duration ΔT, shown in Fig. 6. Sequential poses, associated with images, are connected with a combination of factors comprising inertial measurement unit (IMU) and optical flow constraints. With only these factors, the system performs VIO. In addition, we add sparse (BRISK) features to the pose graph as projection factors connecting to known feature locations in the sparse map. There are several ARTags [27] located on the Astrobee dock, and these are treated similarly to the sparse features, with the ARTag pose initialized at the time of first viewing. We

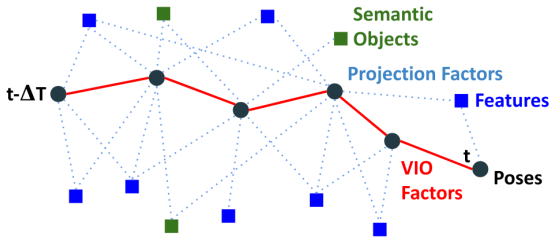

Fig. 6: Visualization of localization pose graph.

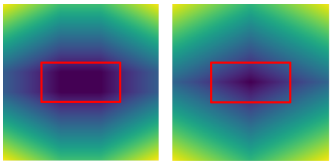

Fig. 7: Visualization of the sum of the x and y costs at different m_α positions using different buffer sizes. The detection bounding box is shown in red.

handle semantic factors similarly as well. When a detection is associated with an object in the map, creating the tuple $(m_j, d_{k,n}, p_k)$, we add a projection factor between the pose and object using the detection and continue to optimize the graph. The additional computational burden of our method is therefore quite minimal, since there are relatively few semantic object factors added to the optimization compared to sparse feature factors.

We make several modifications to the semantic factors in order to further improve performance. Firstly, the standard deviation $\sigma = (\sigma_x, \sigma_y)$ of the noise model is scaled by the size of the detection bounding box $\text{bbox}(d_{k,n})$. We do this because we expect noise in the detector of the order of magnitude of the bounding box size. In addition, if an object is partially in view, the bounding box center may not necessarily align with the object's actual centroid. In order to combat this, we set a threshold τ as a function of the bounding box size where no cost is applied. Finally, we scale the uncertainty inversely with the number of matched detections, motivated by the intuition that a larger number of associations suggests stronger confidence in those associations, as well as more strongly constrains the graph. A visualization of the cost, varying the position of m_α with respect to a detection, under different τ is shown in Fig. 7. Formally,

$$\begin{aligned} \sigma &= \text{bbox}(d_{k,\beta}) R / N_{\text{match}} \\ E &= \max\left\{ \left| K p_k^{-1} m_\alpha - \text{pos}(d_{k,\beta}) \right| - \tau \text{bbox}(d_{k,\beta}), 0 \right\} \end{aligned} \tag{1}$$

where R is a tunable parameter and E is the cost. In practice, instead of the max we use two Sigmoid Linear Units [28], which are a smooth approximation with a continuous Jacobian, making the graph optimization more robust.

We now have only the data association problem left to solve. We firstly define a cost $C_{\alpha\beta}$ where for the association of object m_α with detection $d_{k,\beta}$, we have

$$C_{\alpha,\beta} = \left\| (K p_k^{-1} m_\alpha - \text{pos}(d_{k,\beta})) / \text{bbox}(d_{k,\beta}) \right\|_2 \tag{2}$$

We can then formulate the data assocation optimization

$$\begin{aligned} &\underset{J_k}{\textbf{minimize}} \quad \sum_n C_{j_{k,n},n} \\ &\textbf{subject to} \quad j_{k,a} \neq j_{k,b}, j_{k,a} \leq |M|, j_{k,a} \in \mathbb{Z}^+ \ \forall \ a, b \end{aligned} \tag{3}$$

where $J_k = \{j_{k,1}, \ldots, j_{k,N}\}$. In other words, for a given image k, we want to find the unique assignment J_k between detections and objects that minimizes the sum of the distance between the paired detections and objects. Note that we scale the distance by the size of the detection bounding box. The optimal solution to this problem is given by the Hungarian algorithm [29] operating on the cost matrix C with elements $C_{\alpha,\beta}$.

We note that the number of mapped objects and detected objects may differ, so there may be unassigned objects or detections. Therefore, we create virtual objects or detections with an equal large distance to eachother, padding C with some large constant cost Γ to make it square. In addition, we set elements of C exceeding a threshold to Γ. We then simply ignore the resulting assignments with cost Γ, thereby efficiently thresholding the cost a given association can have.

D. System Design

As described in [1], the majority of the Astrobee autonomy stack is built on ROS and runs on the mid-level processor (MLP). In order to avoid increasing computation demand on this processor, we run the detector on the high-level processor (HLP) under Android, implemented in TensorFlow Lite. Images are sent from the MLP to the HLP and detections are returned over the internal network. We tested our model on a development board with the same processor as the HLP, and found inference on our model to run around 4 Hz. For the experiments presented here, algorithms were run offline, but the detection rate was conservatively artificially set to 1 Hz.

IV. Datasets

We tested on 5 separate datasets, spanning across three different days several months apart in the Japanese Experiment Module (JEM) on the ISS. These datasets are detailed in Table I, where the dataset number (1, 2 or 3) indicates the day. Example images taken on each day from a similar location are shown in Fig. 8. Note the variations in lighting and rack configurations, as well as object locations. All training images for our object detector were taken from dataset 1. For testing, we built a sparse map and semantic map from datasets D1 and D2a in order to compare the methods using the same data, using the high quality poses from our offline SfM pipeline detailed in Sec. III-B. Table I shows which maps are used for localization for which datasets in our experiments. For computing error from ground truth, we use the SfM poses as well.

V. Results

For all experiments, we ran the localizer in VIO mode, that is, not including factors from sparse map features or semantic objects. We also tested on the same datasets using sparse mapping factors, semantic factors, and both. For these cases, the sparse maps and semantic maps are built from the datasets indicated in Table I.

Dataset	Duration (min:sec)	Length (m)	Map
D1	31:23	28.7	$\mathcal{M}_{2a}$
D2a	12:33	15.9	$\mathcal{M}_1$
D2b	4:45	7.5	$\mathcal{M}_1$
D3a	10:56	16.2	$\mathcal{M}_{2a}$
D3b	7:43	22.8	$\mathcal{M}_{2a}$

TABLE I: Datasets used for Evaluation

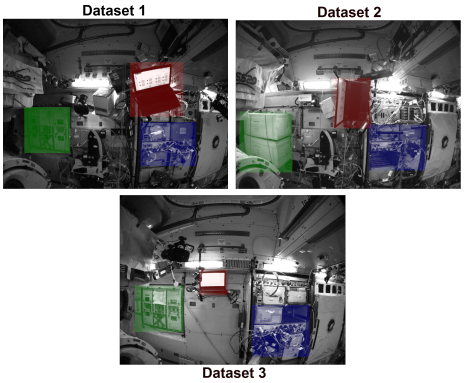

Fig. 8: Images taken from a similar pose from each day represented in our datasets. Some notable differences are highlighted.

A. Qualitative Results

We plot the position errors for three datasets over time in Fig. 9. While orientation errors are not shown for clarity, they follow similar trends. Note that incorporating semantics leads to lower position errors at almost every instant as opposed to simply forward-integrating with VIO. However, semantics do not generally achieve as low an error as sparse features when such feature matches are available. Regions when the localizer is able to register sparse point features are highlighted in the plots. Due to the localizer using sparse maps built under different conditions, it is frequently unable to locate the current image. When this is the case, the error can gradually increase as the localizer relies on VIO. This is particularly evident in dataset 1, where the error grows to be greater at one point than when purely using VIO, likely due to the IMU bias estimation being poor at the time when sparse registration was lost.

By contrast, semantic objects are detected and associated in many more frames, despite using a map built from the same data as the sparse map. Therefore, by incorporating both semantic objects and sparse features we achieve the accuracy of sparse features when they are available, but the semantics increase robustness, decreasing drift when point features cannot be registered.

The error traces for datasets 1 and 2a end earlier when sparse features are not used due to some implementation details of timestamp handling. However, this only occurs once the robot has docked. For computing root-mean-squared-error (RMSE), we simply ignore these portions of the datasets. In addition, note that the starting positions differ

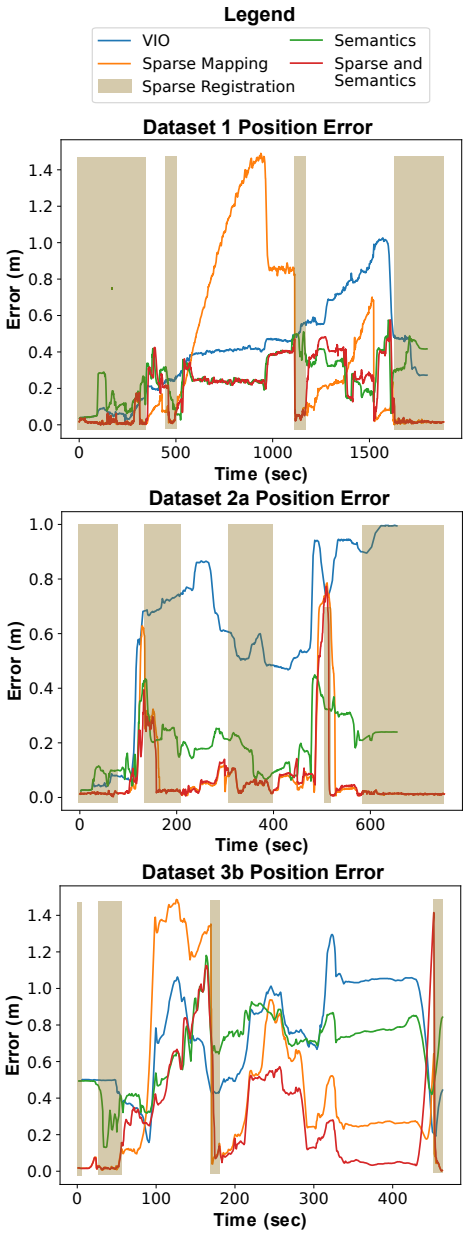

Fig. 9: Position errors over time for different datasets using different localization strategies. The shaded regions show where sparse feature factors were added.

for Dataset 3b depending on the localization method. This occurs because the initial position estimate is not quite accurate, but when sparse features are used it is corrected immediately. When semantic features are incorporated, the localizer corrects itself in the correct direction over time.

B. Quantitative Results

We compute the RMSE error for all datasets, which are plotted in Fig. 10. Dataset 2b has exceptionally large VIO error due to the robot not starting in the dock and beginning motion quickly, causing poor IMU bias estimation and large amounts of drift. This drift is strong enough that semantics are unable to correct effectively before there is sufficient drift for data assocations to begin to be incorrect. This leads to

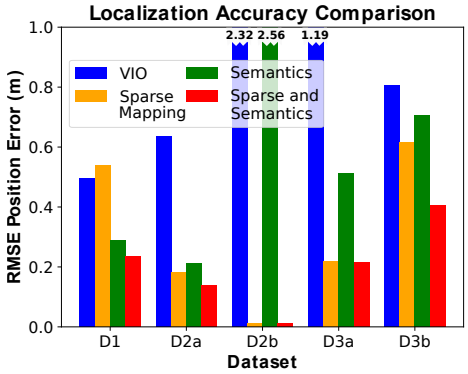

Fig. 10: RMSE position errors over the entire trajectory for each dataset.

large errors for both the VIO and semantics only modes. Incorporating sparse features corrects this error, and adding semantics improves quality further very slightly.

We emphasize that for every dataset, adding semantics improves the RMSE compared to the error from only using sparse features. While in most cases the localizer using semantics alone is unable to achieve as high an accuracy as using sparse features alone, this is unsurprising. A sparse point feature is associated with a specific image point and there are often many of them per-image, while semantic objects are larger, less precisely located, and fewer. However, their greater detection robustness under different conditions leads to the semantic-sparse combination performing most strongly.

In addition, note that datasets 2b, 3a, and 3b demonstrate the efficacy of our method under circumstances where the object detector was not trained on any images in the dataset, nor was the semantic map built using any images in the detector training set. We argue, therefore, that our semantic detector offers excellent robustness to map and lighting changes, even when not explicitly trained on data reflecting those changes. Our method also demonstrates robustness even to changes in the map, since as can be seen in Fig. 8, the locations of objects in the map are not identical between datasets. We expect that performance could be improved further by increasing the variety of images in the training set, as well as using multiple datasets for map construction.

VI. Conclusion

In this work, we have presented methods for object-centric semantic map generation and localization leveraging semantic maps in the context of microgravity. Our map generation method bypasses the data association problem, aside from the final refinement step, by using a voxelized heat map formulation. We have also shown that utilizing semantics in addition to sparse point features leads to increases in localization performance for all of our datasets, and argued that semantics provide a highly robust counterpart to sparse features for absolute localization. In the future, we plan to deploy and test our system in-the-loop on the ISS, enabling greater localization robustness for Astrobee and future free-flying robots.

REFERENCES

[1] L. Fluckiger, K. Browne, B. Coltin, J. Fusco, T. Morse, and A. Symington, "Astrobee robot software: Enabling mobile autonomy on the iss," in *Proc. of the Int. Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS)*, 2018.

[2] M. G. Bualat, J. S. Barlow, J. V. Benavides, B. Coltin, L. J. Flückiger, M. G. Moreira, K. Hamilton, and T. Smith, "Astrobee on-orbit commissioning," in *Proc. IAF Int. Conf. Space Ops*, 2021.

[3] N. J. Fairfield and B. A. Maxwell, "Mobile robot localization with sparse landmarks," in *Mobile Robots XVI*, D. W. Gage and H. M. Choset, Eds., vol. 4573, International Society for Optics and Photonics. SPIE, 2002, pp. 148 – 155. [Online]. Available: https://doi.org/10.1117/12.457439

[4] S. Se, D. Lowe, and J. Little, "Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks," *The international Journal of robotics Research*, vol. 21, no. 8, pp. 735–758, 2002.

[5] H. Bavle, S. Manthe, P. de la Puente, A. Rodriguez-Ramos, C. Sampedro, and P. Campoy, "Stereo visual odometry and semantics based localization of aerial robots in indoor environments," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 1018–1023.

[6] R. Soussan, B. Coltin, V. Kumar, and T. Smith, "Astroloc: An efficient and robust localizer for a free-flying robot," in *2022 IEEE International Conference on Robotics and Automation (ICRA)*, 2022.

[7] B. Coltin, J. Fusco, Z. Moratto, O. Alexandrov, and R. Nakamura, "Localization from visual landmarks on a free-flying robot," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 4377–4382.

[8] S. Leutenegger, M. Chli, and R. Y. Siegwart, "Brisk: Binary robust invariant scalable keypoints," in *2011 International Conference on Computer Vision*, 2011, pp. 2548–2555.

[9] P. Kim, B. Coltin, O. Alexandrov, and H. J. Kim, "Robust visual localization in changing lighting conditions," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 5447–5452.

[10] M. Tan, R. Pang, and Q. V. Le, "Efficientdet: Scalable and efficient object detection," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 10 781–10 790.

[11] A. Gawel, C. Del Don, R. Siegwart, J. Nieto, and C. Cadena, "X-view: Graph-based semantic multi-view localization," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1687–1694, 2018.

[12] R. Mur-Artal and J. D. Tardós, "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras," *IEEE transactions on robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.

[13] S. Lenser and M. Veloso, "Sensor resetting localization for poorly modelled mobile robots," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, vol. 2, 2000, pp. 1225–1232 vol.2.

[14] S. L. Bowman, N. Atanasov, K. Daniilidis, and G. J. Pappas, "Probabilistic data association for semantic slam," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 1722–1729.

[15] R. Anati, D. Scaramuzza, K. G. Derpanis, and K. Daniilidis, "Robot localization using soft object detection," in *2012 IEEE International Conference on Robotics and Automation*, 2012, pp. 4992–4999.

[16] J. Li, D. Meger, and G. Dudek, "Semantic mapping for view-invariant relocalization," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 7108–7115.

[17] Y. Liu, Y. Petillot, D. Lane, and S. Wang, "Global localization with object-level semantics and topology," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 4909–4915.

[18] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison, "Slam++: Simultaneous localisation and mapping at the level of objects," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2013, pp. 1352–1359.

[19] A. Rosinol, M. Abate, Y. Chang, and L. Carlone, "Kimera: an open-source library for real-time metric-semantic localization and mapping," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 1689–1696.

[20] Y. Wu, Y. Zhang, D. Zhu, Y. Feng, S. Coleman, and D. Kerr, "Eao-slam: Monocular semi-dense object slam based on ensemble data association," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 4966–4973.

[21] M. Dor and P. Tsiotras, "ORB-SLAM applied to spacecraft non-cooperative rendezvous," in *2018 Space Flight Mechanics Meeting*, 2018, p. 1963.

[22] S. Nolet, "The spheres navigation system: from early development to on-orbit testing," in *AIAA Guidance, Navigation and Control Conference and Exhibit*, 2007, p. 6354.

[23] S. Mitani, M. Goto, R. Konomura, Y. Shoji, K. Hagiwara, S. Shigeto, and N. Tanishima, "Int-Ball: Crew-supportive autonomous mobile camera robot on ISS/JEM," in *Proc. IEEE Aerospace Conf.*, 2019, pp. 1–15.

[24] G. Blott, M. Takami, and C. Heipke, "Semantic segmentation of fisheye images," in *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, 2018, pp. 0–0.

[25] E. D. Cubuk, B. Zoph, J. Shlens, and Q. V. Le, "Randaugment: Practical automated data augmentation with a reduced search space," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2020, pp. 702–703.

[26] F. Dellaert, "Factor graphs and gtsam: A hands-on introduction," Georgia Institute of Technology, Tech. Rep., 2012.

[27] M. Fiala, "Artag, a fiducial marker system using digital techniques," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 2. IEEE, 2005, pp. 590–596.

[28] S. Elfwing, E. Uchibe, and K. Doya, "Sigmoid-weighted linear units for neural network function approximation in reinforcement learning," *Neural Networks*, vol. 107, pp. 3–11, 2018.

[29] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.