Robust Vision-based Multi-spacecraft Guidance Navigation and Control using CNN-based Pose Estimation

Becktor, Jonathan Binner; Seto, William; Deole, Aditya ; Bandyopadhyay, Saptarshi ; Rahimi, Niyousha ; Talebi, Shahriar ; Mesbahi, Mehran ; Rahmani, Amir

Published in:
Proceedings of 2022 IEEE Aerospace Conference

Link to article, DOI:
10.1109/AERO53065.2022.9843396

Publication date:
2022

Document Version
Peer reviewed version

Robust Vision-based Multi-spacecraft Guidance Navigation and Control using CNN-based Pose Estimation

Jonathan Becktor
Technical University of Denmark
2820, Lyngby, Denmark
jbibe@elektro.dtu.dk

William Seto
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak GroveDrive, Pasadena
California 91109, USA
william.seto@jpl.nasa.gov

Aditya Deole
University of Washington
3940 Benton Lane NE, Seattle
Washington 98195, USA
adityad@uw.edu

Saptarshi Bandyopadhyay
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak GroveDrive, Pasadena
California 91109, USA
saptarshi.bandyopadhyay@jpl.nasa.gov

Niyousha Rahimi
University of Washington
3940 Benton Lane NE, Seattle
Washington 98195, USA
nrahimi@uw.edu

Shahriar Talebi
University of Washington
3940 Benton Lane NE, Seattle
Washington 98195, USA
shahriar@uw.edu

Mehran Mesbahi
University of Washington
3940 Benton Lane NE, Seattle
Washington 98195, USA
mesbahi@uw.edu

Amir Rahmani
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak GroveDrive, Pasadena
California 91109, USA
amir.rahmani@jpl.nasa.gov

***Abstract*— In this paper, we present an end-to-end simulation framework for tracking an uncooperative Target spacecraft in Low Earth Orbit using a CubeSat-class Ego spacecraft outfitted with a camera. Currently, capturing high-fidelity realistic images in space for this scenario is difficult and exorbitantly expensive. Therefore, we developed a framework to simulate the spacecraft orbits in Basilisk software and generate high-fidelity realistic images of spacecraft in Unreal Engine, including the effects from Sun, Earth, Moon and stars.**

The Ego spacecraft uses cameras to capture images of the uncooperative Target and estimates its position and attitude using a CNN based 6DOF pose estimation pipeline, eliminating need for large SWAP-C(Size, Weight, Power and Cost) sensors like LIDAR or reliance on inter-spacecraft communication, This CNN, which is motivated by ESA's Pose Estimation challenge of 2019, is trained using simulated data from our end-to-end simulation framework. We compare the performance of two distinct CNN-based algorithms for pose estimation along a nominal trajectory. In presence of non-Gaussian modeling uncertainties, the state-dependent estimation error is characterized with a quadratic upper-bound. The quadratically-bounded error can be used by a robust controller to maneuver Ego spacecraft to track the uncooperative Target.

TABLE OF CONTENTS

978-1-6654-3760-8/22/$31.00 ©2022 IEEE

1. INTRODUCTION

Multi-spacecraft missions can be used for a wide variety of science applications like distributed sensors for Earth Science and interferometry for Astrophysics, and technology applications like in-space assembly, debris removal, and docking with uncooperative targets. A common challenge in many of these application is tracking an uncooperative spacecraft or space-debris (called *Target*) using one a spacecraft or smallsat (called *Ego*).

The traditional approach to multi-spacecraft guidance, navigation and control involves using a combination of sensors like differential-GPS, laser/reflector meteorology, gyros, and inter-satellite communication, for relative pose (position and attitude) estimation and a control framework for precise formation keeping and reconfiguration. But many of these active sensors cannot be put on an uncooperative Target.

Moreover, high-quality Electro Optic (EO) cameras have become readily available for close proximity operation and relative pose (position and attitude) estimation of spacecraft. Vision based estimation can also augment differential-GPS, which is usually not reliably available beyond LEO and requires constant inter-spacecraft communication. EO camera sensors are compact, space tested, and can provide a rich set of information, making them an ideal sensor for multi-spacecraft smallsat missions.

Superior computational capabilities onbaord the Ego spacecraft enable extraction of relevant pose (position and attitude) estimates from the images of the Target spacecraft using appropriate estimation filters. While older methods have relied on simple computer vision algorithms along with carefully placed fiducial markers, new powerful onboard computers

allow the use of a new set of vision-based relative estimation tools based on Artificial Intelligence and Machine Learning, specifically Convolutional and Recurrent Neural Networks (CNN and RNN). In the past couple of years, a large body of research has demonstrated the superior performance of such CNN/deep-learning based relative spacecraft pose estimation compared to classical techniques, drawing inspiration from robotics research and applications [1], [2], [3], [4]. The ESA Pose Estimation Challenge in 2019 helped in the advancement of a number of these vision-based pose estimation algorithms [5]. In fact, vision+CNN based relative pose estimation is the only approach that works for proximity operations around passive/uncooperative targets.

While these methods are powerful, they rely on training a neural network with a database of synthetic images that cover all expected views from the perspective of the onboard camera during the life-time of the mission. As such, these CNN/RNN act as blackboxes that receive images and output an estimate of the pose. This blackbox approach, along with reliance on synthetic training data that might be different from what might be encountered in orbit, can lead to unreliable estimates that are far off from reality and do not follow a usually assumed Gaussian error model. Most research on mitigating this issue has been focused on "explainable AI", and essentially trying to characterize this general estimation using neural networks and potential reasons for failure to produce a quality estimate. This focus on separating estimation (relative navigation) and control can lead to unforeseen stability issues that can arise from failure of the ML (CNN/RNN) based relative pose estimation (both due to its blackbox nature as well as the insufficient ground based training data); which can excite the otherwise stable formation flying controller such that the whole closed loop becomes unstable.

In this paper, we lay the foundations to address this key Guidance Navigation and Control (GNC) challenge that must be solved for the aforementioned missions to succeed by:

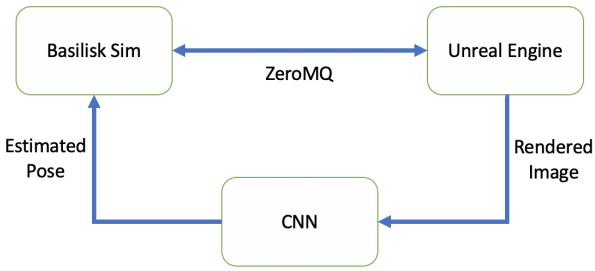
1. Setting up an end-to-end simulation pipeline to simulate the spacecraft orbits and generate high-fidelity realistic images for closed-loop GNC testing,
2. Quantitatively characterizing the error of CNN/RNN-based visual estimation techniques, and
3. Incorporating the error estimates in a robust formation controller; hence providing a robust end-to-end pipeline for control of multi-spacecraft missions.

Our Robust Data-driven Vision-based multi-Spacecraft (RDVS) Guidance and Control framework is a robust autonomy framework for multi-spacecraft missions that incorporates vision+CNN based relative navigation, is robust to failures and errors in the estimation pipeline, and provides theoretical guarantees on the performance of mission for formation keeping and reconfiguration applications. We characterize the performance of vision+CNN based relative pose estimation and use control theoretic techniques to combat the failure modes of vision based relative navigation. In this paper, we present the first-ever end-to-end control framework, built from ground up, that is robust to failures of vision+CNN based relative navigation for multi-spacecraft missions.

2. End-to-End Simulation Pipeline

In this section, we describe the architecture and details of the software implementation of the simulation pipeline used to validate our methods.

Figure 1: Overview of simulation pipeline with each software process. The arrows illustrate how the processes communicate.



Pipeline - Spacecraft Dynamics

Basilisk is a software framework for spacecraft simulations developed by the University of Colorado [6]. The framework has comprehensive features such as faster-than realtime simulation, and hardware-in-the-loop support. Basilisk was chosen for its ease of use and reconfigurability, as it has a fully-featured Python interface. An entire simulation is executed as a single script, which aids in rapid prototyping. New control laws are simply implemented as new Python modules that Basilisk can integrate internally, while having fine grained control of the framework in Python allows us to expose the state of the simulation in order to integrate external modules such as the CNN and have them run in lockstep with the rest of the simulation. Additionally, we retain the fidelity of dynamics and speed of execution as Basilisk is implemented in C/C++.

Pipeline - Image Rendering

To support the development and testing of the CNN based pose estimation algorithms, we utilize Unreal Engine to render a 3D visualization of the scene based on the simulation state. The software allows us to render high fidelity photo-realistic scenes by providing us the ability to place the celestial bodies precisely, along with the lighting source, and model the atmosphere as well. Additionally, we have fine control over the rendering settings and image filters, which allows us to add noise and perturb the observations to enrich the CNN training datasets.

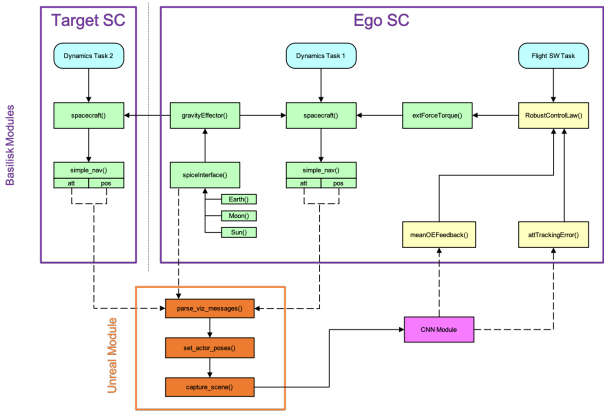
To setup the scene in Unreal Engine, we create a "level", which is basically a collection of Actors with their associated properties and locations. Then at runtime, instead of programming custom game logic and functions to move the actors, we instead utilize the scripting interface to kick off a thread that listens to external updates, described in more details below.

Pipeline - Integration

This subsection describes how three software processes: the spacecraft dynamics simulation, the renderer/visualization, and the CNN interact in the full simulation pipeline.

Figure 1 shows the general software architecture. The Basilisk simulation process and the Unreal Engine game thread communicate via a socket-based library called ZeroMQ. The reason we chose ZeroMQ was because the Basilisk framework natively ships with its own visualization tool, and its communication model is based on ZeroMQ, so we wanted to reuse that part of the library. The communication model is a streaming one, in which the Basilisk process

Figure 2: The end-to-end simulation pipeline, where all intermediate functions are shown.



pushes out the entire simulation state, which includes the position, velocity, and orientation of all the celestial bodies at every timestep over the socket. On the other side, the visualization tool listens to these state messages and updates the locations of each of the celestial bodies to be rendered. In our project, we are replicating how Basilisk would communicate with its own visualization tool, but with Unreal Engine instead.

Next, to integrate the CNN process with the rest of the pipeline, we use an indirect form of communication, in which each process waits for the existence of a file generated as output by another process before continuing. First, Unreal Engine takes the current simulation state from Basilisk, updates the actor locations, and then renders an image from the viewpoint of the Ego spacecraft. Then, the CNN process waits for this image to be written to file and then opens it and runs detection. Finally, the CNN process writes the detected pose to a file, and then the Basilisk process picks it up and incorporates the estimate as input to the attitude control law to point the Ego spacecraft to the Target spacecraft. While simplistic, this approach works well for our rapid prototyping, and accommodates running pose estimation at a lower rate (1 Hz) compared to the simulation rate (10 Hz).

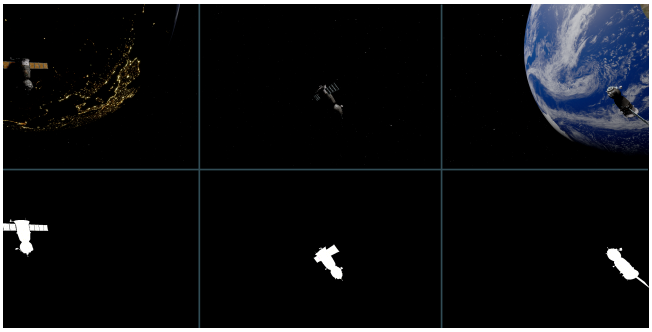
3. CNN-BASED POSE ESTIMATION USING KEY-POINTS AND PERSPECTIVE-N-POINT

In this section a framework inspired by the pose and position estimation pipeline from [7] will be explored. This pipeline consists of three separate operations, target segmentation, key-point estimation, and attitude and position estimation, where mask-rcnn [8], HRNets [9], and perspective and point algorithm [10], [11], [12] is used respectively. Unreal Engine 4 is used for data generation which approximates the actors in a LEO (Low Earth Orbit) setting.

Segmentation

The initial operation is to find the Target from the view of the EGO. This is done by the mask-rcnn. neural network architecture. This is a mask based segmentation neural network using a resnet backbone. We use the PyTorch implementation of Mask-RCNN. Given an input image it predicts: a mask, bounding box and a class for all predicted objects in the input, in this case either the Target or background. The resulting masks are used to generate the region of interest, by relaxing

Figure 3: Samples of segmentation training data generated by Unreal Engine in LEO with varying sun angles, Earth/Moon/Target positions and orientations.



the containing bounding box to ensure the entire surface of the Target is within. This allows us to use high resolution images for the following operations as we can now crop to the region of interest and discard unnecessary data. The network is trained with the AdamW [13] optimizer, with Cosine Annealing Warm Restarts [14] learning rate scheduler with an initial learning rate of $1e^{-}4$, batch-size of 8 and trained on a RTX-3090. The loss follows the standard implementation:

$$L = L_{cls} + L_{box} + L_{mask}$$

as described in the mask-rcnn paper [8]. The generated dataset consists of 24,000 generated images in LEO. Each dataset has been sampled uniformly from a set of uniformly distributed relative sun angles, positions and orientation of earth, moon, Target and Ego; see Figure 3.

Key-points

Given a segmentation, it is now possible to crop the image to the region of interest and resize it accordingly. The approach here is to have the network predict unique key points of the Target such as the corners of the solar array, the satellite disc etc. The key-point network uses the HRNet architecture (Figure 6). Again the network is trained with the AdamW [13] optimizer, with Cosine Annealing Warm Restarts learning rate scheduler with an initial learning rate of $1e^{-}5$, batch-size of 16 and trained on a RTX-3090. The network uses the minimization of the mean squared error,

$$L_{\text{MSE}} = \frac{1}{N}\sum_{i}^{N}(y - f(x))^2$$

for N samples, where y is the output of the model,y is the target, in order to characterize the predicted key-points given the input image. The ground truth for each key-point is generated as 2D normal distribution with mean equal to the ground truth location, and a standard deviation of 1-pixel. The dataset consists of 83,000 images, generated by Unreal Engine in LEO with varying sun angles where the sun never is directly visible, and Earth/Moon/Target positions/orientations. The relative distance between EGO and Target varies from 15-45 meters. Thus when the region of interest is selected in the original image, as the distance increases, the pixel density is reduced at a quadratic rate which can be seen in Figure 5.

Perspective-n-Point

Given the predicted key-points and the known Target object points, we can compute the transformation required to project

Figure 4: The full pipeline for estimation of position and orientation.

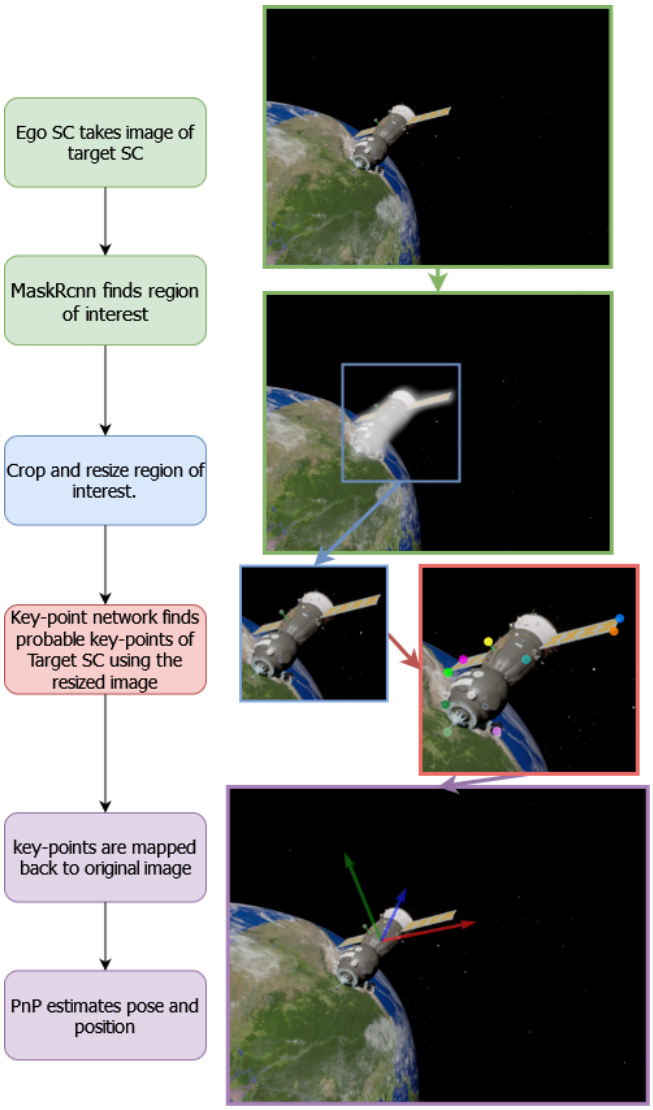


Figure 5: Samples of key-point training data generated by Unreal Engine in LEO with varying sun angles, Earth/Moon/Target positions and orientations. The cropped and resized versions have lower resolutions given the distance to Target here approximately at 20, 30, 40 meters.

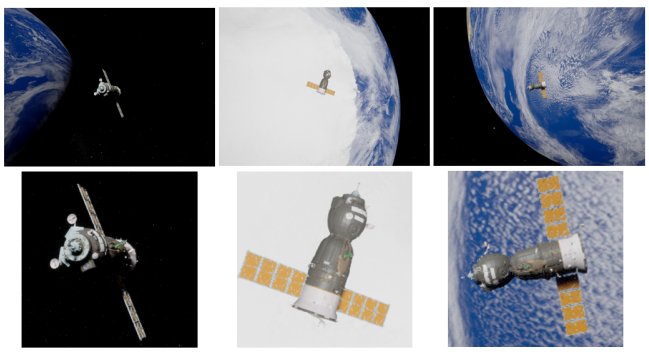
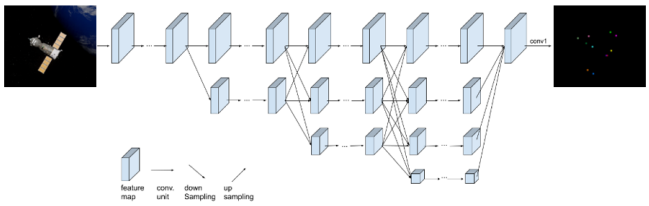


Figure 6: HRNet network structure, original figure from [7].



object-points onto key-points. The initial estimation of the position and orientation is found by the OpenCV [15] implementation of the RANSAC PnP solver using the SQPnP [12] kernel. This allows for detection of outlying key-points and provides an initial attitude and position. This is fine-tuned using an iterative Levenberg-Marquandt-based (LM) optimization. By minimizing the re-projection error–that is the sum of squared distances between the observed projections key-points and the projected object Points–this optimization procedure provides a position and orientation of the desired Target in ego frame.

Pipeline Results

The position loss is the ℓ_2 norm between prediction (t_p) and ground truth (t_{gt}) translation:

$$L_{\text{T}} = \sum_{i}^{m} \frac{||t_{\text{p}}^{(i)} - t_{\text{gt}}^{(i)}||}{||t_{\text{gt}}^{(i)}||} \tag{1}$$

The pose loss is the angle between the predicted quaternion (q_{p}) and the target quaternion (q_{gt}) measured in Radians, as seen here,

$$L_{\text{R}} = 2 \cdot \arccos(|q_p^{(i)\top} q_{gt}^{(i)}|) \tag{2}$$

The total loss can be found with,

$$L_{\text{total}} = E_T + E_R. \tag{3}$$

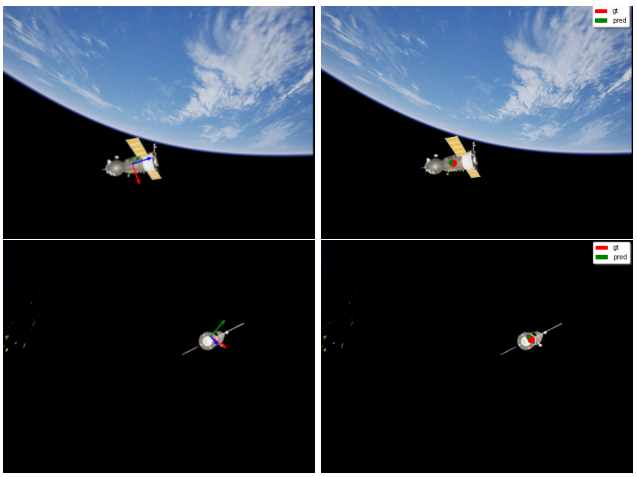
The testing set is created separately and consists of samples within a 15-35 meter distance between Ego and Target, sun angles are uniformly sampled on a sphere around Ego with the exception of in the direct field of view of Ego.

Table 1: Results of the total loss L_{total} on two separate test sets are made with 1MP and and 12MP images respectively.

Metric	1 MP	12 MP
$CI: 95\%$ μ_{E_T} (Meters)	$0.037m$	$0.020m$
$CI: 99\%$ μ_{E_T} (Meters)	$0.064m$	$0.038m$
$CI: 95\%$ μ_{E_R} (Radians)	0.149rad	0.103rad
$CI: 99\%$ μ_{E_R} (Radians)	0.192rad	0.146rad

From 1 we observe that there is a clear benefit in using the higher resolution images. In fact, the pipeline provides a precise position and a usable attitude estimate for the input. The most common error in the pipeline occurs when the key-point network mirrors certain key-points or does not find a sufficient number of them. The pipeline provides a certainty score along with each key-point so if the total certainty is low this will be passed along to the controller along with the attitude and position.

Figure 8: Two different samples of the data generated using Unreal Engine showing the target (Soyuz spacecraft) superimposed with, a) the pose estimation, b) the position estimation in green versus the ground truth in red.



(a) (b)

4. CNN-BASED DIRECT IMAGE TO POSE ESTIMATION

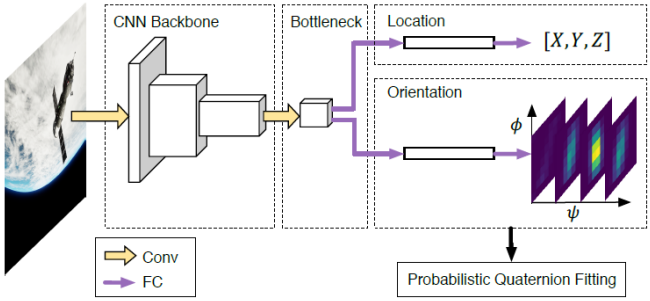
Herein, we present an adaptation of the network architecture developed by Proenca et al. [1] for known uncooperative spacecraft tracking problems on low earth orbit (LEO). This network is based on a widely used ResNet architecture with pre-trained weights as its backbone which provides monocular pose estimation of the target spacecraft from a single image. More precisely, the 3D location estimation is a regression branch with two fully-connected layers with the total loss function

$$L_{\text{total}} = \beta_1 L_T + \beta_2 L_{\text{ori}}, \tag{4}$$

with L_{T} defined in equation 1. Furthermore, $\{\beta_1, \beta_2\}$ represents the loss-weights to be fine-tuned. The orientation loss L_{ori}, on the other hand, is designed to represent a probabilistic orientation estimation via soft classification and modeling uncertainties (due to orientation ambiguity) as a mixture model. This is shown to outperform the direct orientation regression with the error defined in the previous section in equation 2. Yet, the latter cost is still useful in the testing phase (see [1] for further details).

In this work, we have utilized Unreal Engine Simulator to generate training datasets. In this scenario, the uncontrolled target travels in a fixed LEO and the Ego spacecraft follows

Figure 9: Simplified architecture of CNN as depicted in [1]. Where FC is a fully connected layer and Conv is the convolutional layer.

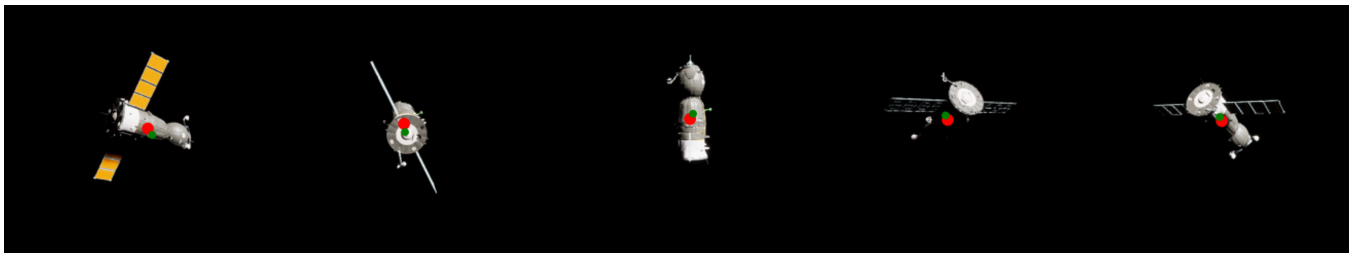


in a relative fixed orbit around the target in its Hill's frame. The fixed orbital distance between the two spacecraft ranges between 10m to 40m. To generate training data, we first develop a nominal controller for tracking purposes. The camera is fixed along the x-axis of Ego's body-frame. Once the target appears in the Ego frame, the nominal controller steers the Ego's attitude to place the target at the center of the Ego frame. We explicate the details of developing these nominal trajectories in the next section. Using this nominal controller, we generate 100 trajectories starting from random initial poses, for different sun angles, over a time span of 30 seconds. Images of the target are then taken from the Ego spacecraft at known relative poses on the nominal trajectories at a rate of 2Hz. Additional images are also generated from randomised target locations in a fixed LEO, on the sun side of Earth, while Ego and target attitudes are perturbed. These additional images help with exploring edge cases. Figure 8 illustrates an example of the generated data that are used for training, superimposed with the estimation. Also, examples of the relative position estimation of the target with different relative orientation is illustrated in Figure 7.

Table 2: Performance of CNN for test set of 3000 images

Metric	1 MP
$CI: 95\%\ \mu_{E_T}$ (Meters)	$0.046m$
$CI: 99\%\ \mu_{E_T}$ (Meters)	$0.046m$
$CI: 95\%\ \mu_{E_R}$ (Radians)	0.487rad
$CI: 99\%\ \mu_{E_R}$ (Radians)	0.494rad

Figure 7: Samples of the relative position estimation of the target (Soyuz spacecraft) using the adapted CNN, zoomed in to better illustrate the ground truth in red versus the estimation in green.



5. PERFORMANCE OF CLOSED-LOOP ROBUST CONTROLLER

In general, CNN-based relative pose estimation algorithms rely on training a neural network with a database of synthetic images that cover all expected views from the perspective of the onboard camera during the lifetime of the mission. To characterize the error of such CNN-based pose estimation, we utilize a dense sampling-based approach and define the regions around the training data for which a learned perception map can be used safely. By considering all variables that affect the perception error in the entire environment, the data complexity exponentially increases. Separating estimation and control is the primary cause of this issue as we ignore the details of the operating regions. Instead, we propose a pipeline to build the perception error profile for a specific nominal trajectory used during the operation. Similar approaches has been used in [16].

The variables that affect the perception error include environmental factors such as changes of the sun angle, reflective surfaces on the target, or reflecting light from the earth. The camera's light exposure and the position of other celestial bodies may also affect the perception error, inducing conditions like bloom or loss of sight. The two most problematic factors that lead to higher estimation error are the sun angle and the geometry of the target. An example of the effects of extreme sun angles on quality of estimation is illustrated in Figure 10.

The variation in output due to illumination, as seen in Figure 12 is inherent in the estimation due to prominent features being occluded. From this experiment, we can extract sun angles, for which the estimation is reliable.

The effects of the target geometry are studied by testing the neural network outputs for randomized target rotations while it is aligned at the center of the camera frame and the sun is located at a reliable spot behind the ego. We pick this setting so that no error induced by lighting condition or ego's attitude would affect the position estimate. Here as seen in Figure 11, due to asymmetric geometry, rotation of target leads to deviation of output from the expected value. There are cases where features are hidden, and body axes cannot clearly be distinguished, which leads to higher estimation error. To model error due to target state we isolate the errors induced by rotation and illumination and keep the sun angle and target pose as constant.

Finally, to establish that the perception error is systematically dependent on the relative state, we demonstrate the min, max and mean of the error (Figure 13) for randomly selected samples in which the target is placed at a certain distance from the ego. As it is shown, both the mean and max of the error increases as the target is further away from the ego in the camera frame.

This setup leads to building a composite error model that is dependent on relative target states in camera frame and environment conditions like sun angle. Errors are also induced due to background fluctuations and noise induced due to camera sensor. To build a generalized error model we need to exhaustively sample over the entire state space and over all the variables mentioned in this section that add estimation error. This is computationally expensive and does not exploit dynamics of the ego-target problem. We instead sample over a nominal trajectory similar to [17], and build the error model around this trajectory.

In what follows, we lay down the details of how we collect the training data from a nominal trajectory and estimate the perception error bounds.

We consider a tracking problem in which the main uncertainty comes from the image sensor, generating observations that capture information about the state of a target relative to an ego spacecraft. Using the neural network structures presented in Sections 3 and 4, a specific (trained) "perception map" is developed to recover the actual relative state of the system from the image data on the fly. Particularly, we first define the relative position $x(t) \in \mathbb{R}^3$ (of the Target with respect to the Ego's position) as

$$x(t) := x_{\text{target}}(t) - x_{\text{ego}}(t); \tag{5}$$

in a passive elliptical orbit defined by nonlinear Clohessy-Wiltshire dynamics [18], with $x_{\text{target}}(t)$ and $x_{\text{ego}}(t)$ denoting their respective positions in the Earth fixed frame. Along this orbit, we consider a sensory observation $z(t) \in \mathbb{R}^{m\times m}$ as

$$z(t) = \mathcal{Q}(x(t), q^E(t)) \tag{6}$$

that represents the measurement of this relative position in the Ego's camera frame, where $q^E(t)$ denotes the Ego's attitude in the Earth fixed frame in Modified Rodrigues Parameters (MRP), and $\mathcal{Q} : \mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}^{m\times m}$ is a nonlinear and potentially quite high dimensional mapping. For simplicity of presentation, we assume the Ego's body frame and its camera frame are identical. The task of the observation module is to detect the target and determine its relative position with respect to the ego spacecraft. The camera is fixed on the Ego's body frame facing its principal x-axis. Here, the observations $z(t)$ are the captured images generated by the unknown map $\mathcal{Q}$. The CNN-based pose estimation techniques are then adopted to represent a *perception map* $\mathcal{P} : \mathbb{R}^{m\times m} \to \mathbb{R}^3$ that imperfectly recovers the relative position $x(t)$ in the camera frame; that is,

$$\mathcal{P}(z(t)) = x(t) + e_p(x(t)), \tag{7}$$

in the camera frame, where e_p denotes the perception error that is dependent on the relative position.

Next, consider Ego's attitude dynamics given by

$$\frac{d}{dt}\begin{bmatrix} q^E \\ \omega^E \end{bmatrix} = f(q^E, \omega^E, u)$$

where ω^E are angular rates and q^E represents attitude in modified Rodriguez parameters. To model such a perception map and study its error characteristics, given some initial state $[q_0^E, \omega_0^E]$ and initial relative position $x(0)$, we design a nominal nonlinear controller that generates a nominal trajectory $\{\overline{q^E}(t), \overline{\omega^E}(t), \overline{u}(t)\}_{t=t_0}^{t_f}$. We then collect the initial training dataset $\mathcal{S}_0 = \{({q^E}_d, z_d)\}_{d=1}^{N_0}$ from the nominal trajectory and train the neural network. Then, starting off from this nominal trajectory with error, we define $\eta(t) := q^E(t) - \overline{q^E}(t)$ as the tracking error, and design a robust controller that tracks the nominal trajectory by driving η to zero. This controller is specially robust to the perception errors e_p that result in an imperfect estimation of η, denoted by $\tilde{\eta}$. These trajectories are illustrated in Figure 14 in the image frame. For further details on the nominal trajectory generation and the robust control synthesis, we refer the interested reader to [19].

Figure 10: Samples of the estimation errors under extreme sun angles illustrating the estimation in green against their ground truth in red.

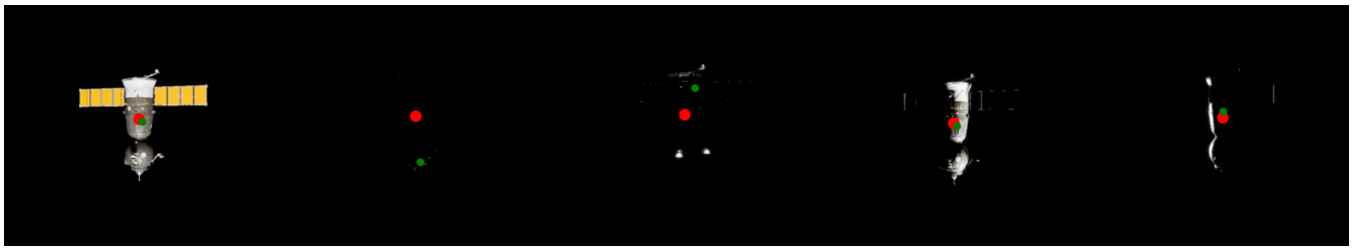


Figure 11: Target rotation leading to measurement error with respect to ground truth at origin

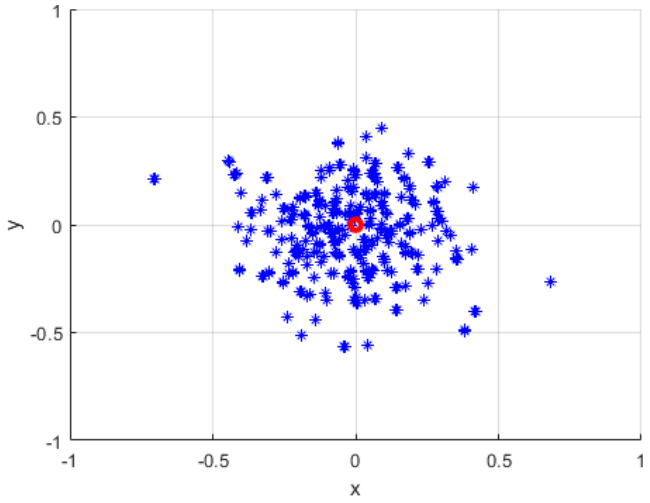


Figure 12: Target aligned with camera center shows error under different lighting conditions

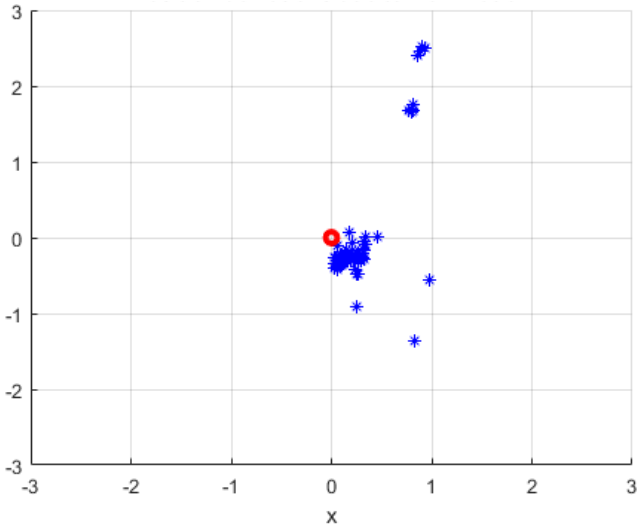


Figure 13: Regional maximum error slope as a function of radial distance of target from center in the image frame

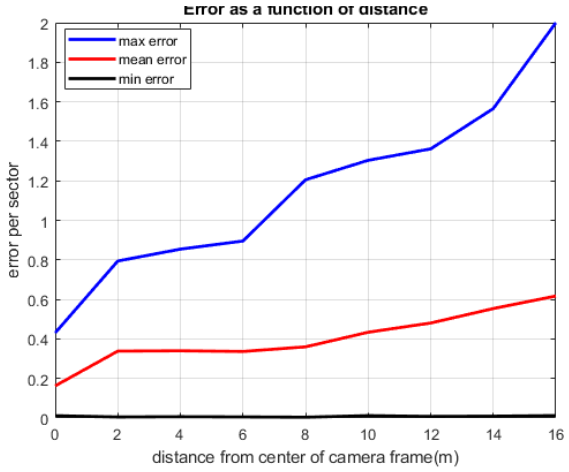


Figure 14: Nominal trajectory in green as seen in the camera frame. For some initial condition which deviate form the nominal as shown in red, we use the robust controller such that the error between the new state and nominal given by η is driven to zero.

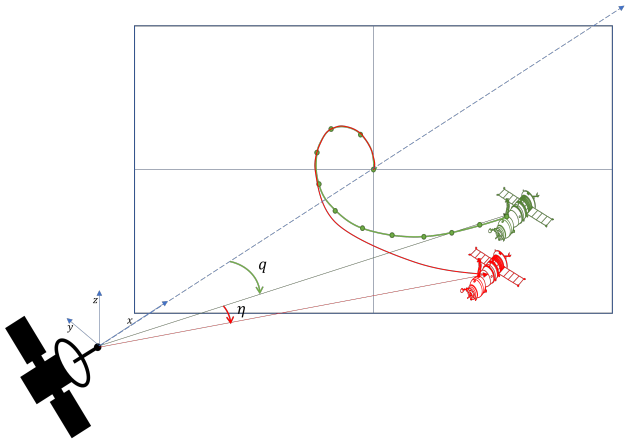


Figure 15: The error profile in the neighborhood of $\overline{q_0^E}$ for a trajectory of 50 seconds given norm of deviation from the nominal at each time.

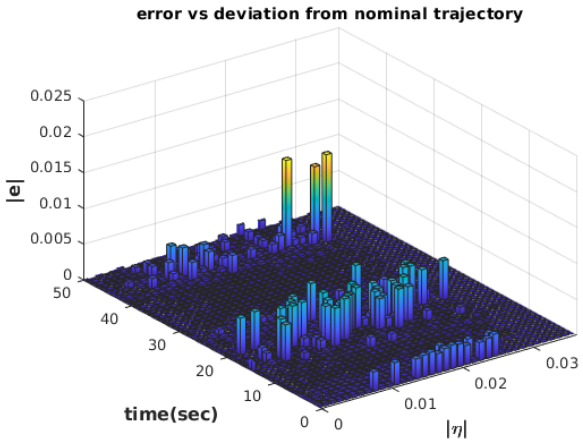
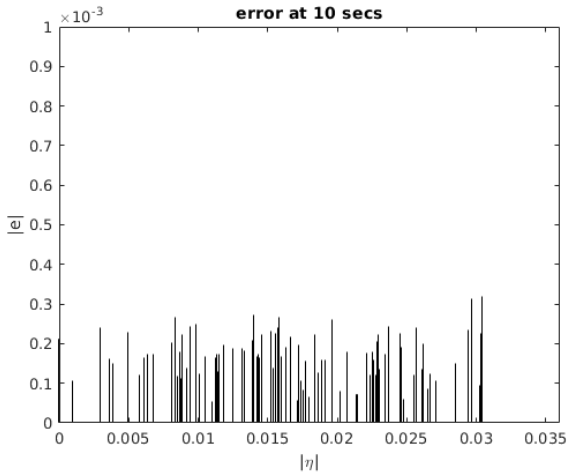


Figure 16: The error profile in the neighborhood of $\overline{q_0^E}$ at t = 10sec.



Our observations as seen in Figure 16 and Figure 15, indicate that the slope bound is dependent on time and deviation from the nominal trajectory. Following these observations an empirical error model is defined as $\|e\| \leq s(t)\|\eta\| + r(t)$. We also observe a residual error due to uncertainties from environment. Figure 17 summarizes the error modelling pipeline, where for a given nominal trajectory used in fine-tuning the CNN, we generate a safe set as the union of regions around each trained data point.

The perception map is now used in the feedback loop with a non linear attitude controller. The ego controller uses the relative position estimates as seen in Figure 18, and estimates attitude set points that align target to the camera center. The controller drives the ego attitude to theses desired setpoints. The design of robust controller for this attitude control is presented in [19].

Figure 17: Pipeline for generating the error profile.

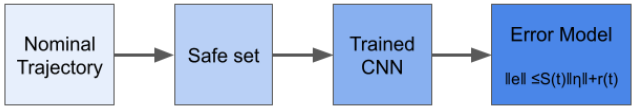
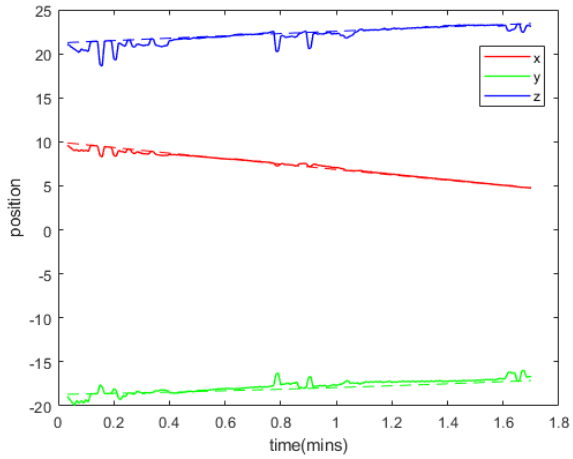


Figure 18: Position estimation along a nominal trajectory, the dashed lines show the ground truth and solid lines show the CNN estimate for relative position of target wrt ego spacecraft.



6. Conclusions

In this paper, we presented an end-to-end simulation pipeline for generating high-fidelity realistic images of multiple spacecraft in space, and implemented a closed-loop guidance, navigation, and control architecture to test the estimation and robust control algorithms. We presented two CNN-based pose estimation algorithms that were tested with a novel robust control framework in this end-to-end simulation pipeline. We envisage this end-to-end simulation pipeline and associated algorithms with help in the development and maturation of robust GNC architecture for multi-spacecraft applications.

Acknowledgments

Part of this research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

References

[1] P. F. Proenca and Y. Gao, "Deep learning for spacecraft pose estimation from photorealistic rendering," 2019.

[2] S. Sharma, C. Beierle, and S. D'Amico, "Pose estimation for non-cooperative spacecraft rendezvous using convolutional neural networks," in *2018 IEEE Aerospace Conference*, 2018, pp. 1–12.

[3] S. D. Sonawani, R. Alimo, R. Detry, D. Jeong, A. Hess, and H. B. Amor, "Assistive relative pose estimation for on-orbit assembly using convolutional neural networks," *CoRR*, vol. abs/2001.10673, 2020.

[Online]. Available: https://arxiv.org/abs/2001.10673

[4] A. Harvard, V. Capuano, E. Y. Shao, and S.-J. Chung, "Spacecraft pose estimation from monocular images using neural network based keypoints and visibility maps," in *AIAA Scitech 2020 Forum*, 2020. [Online]. Available: https://arc.aiaa.org/doi/abs/10.2514/6.2020-1874

[5] S. D'Amico. (2019, July) ESA pose estimation challenge. [Online]. Available: https://kelvins.esa.int/satellite-pose-estimation-challenge/

[6] P. W. Kenneally, S. Piggott, and H. Schaub, "Basilisk: a flexible, scalable and modular astrodynamics simulation framework," *Journal of Aerospace Information Systems*, vol. 17, no. 9, pp. 496–507, 2020.

[7] B. Chen, J. Cao, A. Parra, and T. J. Chin, "Satellite pose estimation with deep landmark regression and nonlinear pose refinement," *Proceedings - 2019 International Conference on Computer Vision Workshop, ICCVW 2019*, pp. 2816–2824, 2019.

[8] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 2, pp. 386–397, 2020.

[9] J. Wang, K. Sun, T. Cheng, B. Jiang, C. Deng, Y. Zhao, D. Liu, Y. Mu, M. Tan, X. Wang, W. Liu, and B. Xiao, "Deep high-resolution representation learning for visual recognition," *TPAMI*, 2019.

[10] X. S. Gao, X. R. Hou, J. Tang, and H. F. Cheng, "Complete solution classification for the perspective-three-point problem," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, 2003.

[11] V. Lepetit, F. Moreno-Noguer, and P. Fua, "EPnP: An accurate O(n) solution to the PnP problem," *International Journal of Computer Vision*, vol. 81, no. 2, 2009.

[12] G. Terzakis and M. Lourakis, "A Consistently Fast and Globally Optimal Solution to the Perspective-n-Point Problem," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 12346 LNCS, 2020.

[13] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *International Conference on Learning Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=Bkg6RiCqY7

[14] ——, "Sgdr: Stochastic gradient descent with warm restarts," in *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*, 2017.

[15] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[16] B. Recht, "A Tour of Reinforcement Learning: The View from Continuous Control," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 2, no. 1, pp. 253–279, 5 2019.

[17] S. Dean, N. Matni, B. Recht, and V. Ye, "Robust guarantees for perception-based control," 2019.

[18] W. Clohessy and R. Wiltshire, "Terminal guidance system for satellite rendezvous," *Journal of the Aerospace Sciences*, vol. 27, no. 9, pp. 653–658, 1960.

[19] N. Rahimi, S. Talebi, A. Deole, M. Mesbahi, S. Bandyopadhyay, and A. Rahmani, "Robust Controller Synthesis for Vision-based Spacecraft Guidance and Control," 2022.

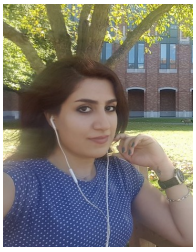
BIOGRAPHY

Jonathan Becktor *received his B.S. and M.S. degrees in Computer Science and Engineering from the Technical University of Denmark, Lyngby, Denmark, in 2015, and 2017 respectively. He was a Visiting Student Researcher at NASA Jet Propulsion Laboratory in 2021, while on his second year of his Ph.D at the Technical University of Denmark, Lyngby, Denmark.*

William Seto *is a Robotics Technologist at NASA's Jet Propulsion Laboratory. He joined JPL in 2017 after receiving his M.S. in Robotic Systems Development from Carnegie Mellon's Robotics Institute. He develops software to enable autonomous capabilities in maritime and terrestrial environments.*

Aditya Deole *is a graduate student in the Department of Aeronautics and Astronautics at University of Washington, working on his Ph.D. degree at the RAIN Laboratory. He received his B.Tech and M.S. in Mechanical Engineering in 2016 and 2020 at Visvesvaraya National Institute of Technology, Nagpur, India and University of Washington, respectively.*

Saptarshi Bandyopadhyay *is a Robotics Technologist at the NASA Jet Propulsion Laboratory, California Institute of Technology, where he develops novel algorithms for future multi-agent and swarm missions. He was recently named a NIAC fellow for his work on the Lunar Crater Radio Telescope on the far-side of the Moon. He received his Ph.D. in Aerospace Engineering in 2016 from the University of Illinois at Urbana-Champaign, USA, where he specialized in probabilistic swarm guidance and distributed estimation. He earned his Bachelors and Masters degree in Aerospace Engineering in 2010 from the Indian Institute of Technology Bombay, India, where as an undergraduate, he co-founded and led the institute's student satellite project Pratham, which was launched into low Earth orbit in September 2016. His engineering expertise stems from a long-standing interest in the science underlying space missions, since winning the gold medal for India at the 9th International Astronomy Olympiad held in Ukraine in 2004. Saptarshi's current research interests include robotics, multi-agent systems and swarms, dynamics and controls, estimation theory, probability theory, and systems engineering. He has published more than 40 papers in journals and refereed conferences.*



Niyousha Rahimi *(S'20) received her B.S. in Mechanical Engineering from Sharif University of Technology, Tehran, Iran in 2016. She received her M.S. in Mechanical Engineering from the University of Washington (UW) in 2018. She is currently pursuing a Ph.D. in Aeronautics and Astronautics at the University of Washington, WA, USA. She was a recipient of Ruth C. Hertzberg Endowed Fellowship in 2018. Her research interests are reinforcement learning, data-driven control, distributed optimization and control. She is also interested in the applications of these areas to robotics and aerospace systems.*

Shahriar Talebi *(S'17) received his B.Sc. degree in Electrical Engineering from Sharif University of Technology, Tehran, Iran, in 2014, M.Sc. degree in Electrical Engineering from University of Central Florida (UCF), Orlando, FL, in 2017, both in the area of control theory. He is currently pursuing a Ph.D. degree in Aeronautics and Astronautics and a M.S. degree in Mathematics at the University of Washington (UW), Seattle, WA. He is a recipient of William E. Boeing Endowed Fellowship, Paul A. Carlstedt Endowment, and Latvian Arctic Pilot–A. Vagners Memorial Scholarships, in 2018-19 at UW, and Frank Hubbard Engineering Scholarship in 2017 at UCF. His research interests are game theory and optimization, variational analysis and networked dynamical systems.*

Mehran Mesbahi *is a Professor of Aeronautics and Astronautics, Adjunct Professor of Electrical and Computer Engineering and Mathematics, and Executive Director of Joint Center for Aerospace Technology Innovation at the University of Washington. He was the recipient of NSF CAREER Award in 2001, NASA Space Act Award in 2004, UW Distinguished Teaching Award in 2005, and UW College of Engineering Innovator Award for Teaching in 2008. He is an IEEE Fellow and co-author of the book* Graph-theoretic Methods in Multiagent Networks *(Princeton University Press, 2010). His research interests are distributed and networked aerospace systems, systems and control theory, and learning.*

Amir Rahmani *is the technical group supervisor of the multi-agent autonomy group at NASA's Jet Propulsion Laboratory. Amir has a Ph.D. from University of Washington in Aeronautics and Astronautics and was an assistant professor of Aerospace Engineering at the University of Miami prior to joining JPL. He has over 15 years of research experience in distributed space systems, formation flying, networked dynamical systems, as well as swarm robotics. Amir is overseeing a number of projects on autonomy technologies for spacecraft and planetary robot teams. He is the NASA STTR subtopic manager on coordination and control of swarms of space vehicles.*