UNCERTAINTY QUANTIFICATION OF A MACHINE LEARNING-BASED POSE ESTIMATION MEASUREMENT MODEL

S. Anand Agrawal[1†*], Brandon A. Jones[2†], and Maruthi R. Akella [3†]; [1]Graduate Research Assistant, [2]Associate Professor, [3]Professor, [†](The University of Texas at Austin & 2617 Wichita St, Austin, TX 78712). [*][aagrawal66@utexas.edu]

Abstract. *The deployment of Convolutional Neural Networks (CNN*s*) within the Guidance, Navigation, & Control (GN&C) suite can enable autonomous satellite proximity operations. CNN*s *have been increasingly utilized as measurement models in navigation filters for the uncooperative spacecraft pose estimation problem. However, requirements for spaceworthy software may necessitate a characterization of uncertainty in the CNN*s*. This work quantifies the uncertainty of a Keypoint Regression Network (KRN) in a pose estimation pipeline using Monte Carlo Dropout (MCD) and analyzes the resulting effects on the pipeline's filtered pose estimates.*

Introduction. Future satellite operations ranging from debris removal[1] and inspection to on-orbit servicing[2] and assembly[3] have driven the need for robust and autonomous space operations in close proximity. A primary requirement for continuous proximity operations is determining the relative pose (position and orientation) from a chaser to a target satellite with high accuracy. These missions often involve a known non-cooperative target, where the satellite of interest is not actively communicating or maneuvering to assist the servicing satellite. Traditional feature-based approaches for this problem, which can perform poorly in the extreme lighting conditions of space, have been supplanted by data-driven methods that leverage deep neural networks, particularly Convolutional Neural Networks (CNNs).[4] Krizhevsky et al.[5] showed that deep CNNs—models with many layers or trainable parameters—can perform object detection effectively when given sufficient data. Sharma et al.[6] were the first to apply this to spacecraft pose estimation by training a CNN with synthetic imagery to both detect a spacecraft and regress its pose. Subsequent works have aimed at improving the performance gap between training on synthetic imagery and inferencing on more realistic space imagery.[7]

Characterizing a CNN's uncertainty is critical to adoption of CNNs inside Guidance, Navigation, & Control (GN&C) systems. There are two main sources of uncertainty when modeling a process: epistemic and aleatoric. Epistemic uncertainty originates from a lack of knowledge of the process, which can be reduced by increasing the amount of data or improving the underlying model and its assumptions; aleatoric uncertainty is a byproduct of the inherent randomness in the process.[8] Predictive or total uncertainty, the combination of epistemic and aleatoric uncertainties, measures the model's prediction confidence with respect to the noise it can and cannot explain.[9] In learning-based spacecraft pose estimation, training on spacecraft imagery that is well-lit may lead to epistemic uncertainty when inferencing on images where the target is poorly illuminated. Aleatoric uncertainty can arise from both real and synthetic imagery from factors including image distortions (e.g., blur and glare) and geometric variations in the target, primarily during synthetic image generation in areas where its 3D model is less detailed.

Quantifying the uncertainty of deep neural networks remains an area of active research. Gal and Ghahramani[10] used dropout layers during training and inferencing to approximate a deep learning model's epistemic uncertainty, which is commonly referred to as dropout variational inference or Monte Carlo Dropout (MCD). Kendall and Cipolla[11] approximated the posterior distribution of the weights to setup a Bayesian neural network and performed MCD to estimate the epistemic uncertainty in a direct pose regression CNN for large scale outdoor scenes. Lakshminarayanan et al.[12] trained multiple networks independently with adversarial learning to produce an ensemble of predictions in which the prediction variance can represent the ensemble's epistemic and aleatoric uncertainty. This method is referred to as ensembling. However, it may be computationally expensive for image-based applications. In Kendall and Gal,[13] a per-sample Maximum A-Posteriori with MCD quantified both aleatoric and epistemic uncertainty. Bramlage and Karg[14] performed deep evidential regression to estimate both epistemic and aleatoric uncertainty for a human pose regression problem.

These works and their applications have utilized datasets that contain real data and are widely available. However, spacecraft pose estimation networks lack a large diversity of real imagery and there is limited treatment of the uncertainty in these networks. Cassinis et al.[15] leveraged the Gaussian kernel in heatmap regression to estimate the (aleatoric) uncertainty of a keypoint regression task for spacecraft pose estimation. Li et al.[16] directly regressed their model's prediction uncertainty to filter keypoint predictions. To the author's best knowledge, these are the only examples of uncertainty quantification within the data driven spacecraft pose estimation literature. Instead, this work prototypes an application of MCD that adds epistemic uncertainty into a spacecraft pose estimation CNN, quantifies that uncertainty and reevaluates it in the context of a noisy measurement model, and demonstrates the quantified uncertainty's impact on a downstream estimation filter. In short, we present an application of MCD to obtain an estimate of predictive uncertainty in the CNN-based measurement model of our spacecraft pose estimation framework.

Background. We address the uncooperative known spacecraft pose estimation problem for monocular imagery using the Cygnus Enhanced cargo resupply craft. A real image of Cygnus is shown in Fig. 1, but the training data as seen in Fig. 2 is entirely synthetic. The Cygnus Body-fixed Frame (BFF) with origin at O^t and the camera BFF with origin at O^c are defined in Fig. 3. The camera BFF represents an imager onboard a chaser spacecraft. Pose estimation seeks to define the position vector that extends from the camera to the target in the camera frame, $\mathbf{r}^c_{c\to t}$, and the quaternion that defines the attitude transformation from the camera BFF to the target BFF, $\mathbf{q}^t_c$. Cygnus provides a useful and challenging target given its symmetry about the xz and yz planes, which can lead to rotational ambiguity. In a later section, we will refer to the Cygnus left solar panel as the one in the positive y^t direction of the Cygnus BFF in Fig. 3. The right solar panel is in the negative y^t direction.

The pose estimation pipeline utilized in this work, which will be referred to as the Vision pipeline for the remainder of this paper, builds upon a three-step process described in Refs. 17 & 18: a bounding box CNN to detect and classify objects, a keypoint or landmark regression CNN to determine the locations of the known keypoints in an image, and a Perspective-n-Point (PnP) static pose solver. Vision's original implementation is established in Ref. 19. Keypoints are points of interest on a target. For example, a CubeSat spacecraft may have keypoints at its vertices, which is the case with the Tango spacecraft from SPEED+.[18] Cygnus' surface keypoints are sourced with a sample elimination algorithm that generates Poisson disk sample sets.[20] This process creates uniformly spaced keypoints and 20 are used here. The projected (20) keypoints in 2D for a synthetic image of Cygnus are displayed in Fig. 2; spatial ambiguity or overlap in the keypoint locations is possible during projection. A Nonlinear Least-Squares (NLS) static pose solver, which also calculates the covariance for each solved pose, and a Multiplicative Extended Kalman Filter (MEKF)[21] to estimate the pose over time have been included in Vision to field a navigation filter.[22] The static pose solver takes predicted keypoint information in the form of azimuth and elevation angles and uses the 2D-3D correspondence of these keypoints to provide an initial pose solution. The CNNs combined with a static pose solver represents a measurement model for the navigation filter, and the Vision overall architecture is shown in Fig. 4. The MEKF dynamics and the NLS measurement model are documented in Ref. 22, and we will present modifications from this starting point.

In Ref. 22, the measurement model's noise covariance matrix is treated as a tuning parameter. We attempt to characterize it with Monte Carlo Dropout. This can be achieved by adding dropout layers to the Keypoint Regression Network (KRN) in Fig. 4. A dropout layer is a stochastic regularization technique[9] that randomly sets a fraction of the inputs into that layer to zero with probability p for each forward pass of the network during training. Such layers can be included for model regularization, preventing overfitting by making the network less dependent on any single input feature.[23] During inference, dropout layers are turned off, leading to deterministic outputs if the activation and weight functions are not stochastic. In practice, MCD offers a means to approximate Bayesian inference without needing to fit probability distributions over the weights. Alternatively, during in-

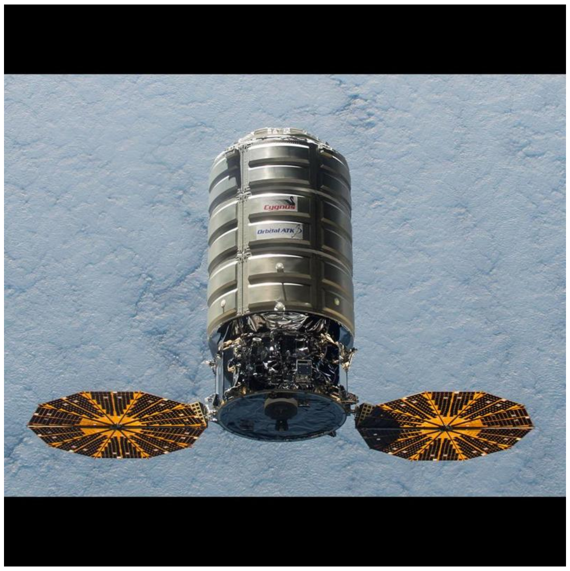


Figure 1. Real Image of Cygnus from the International Space Station (Credit: NASA, December 2015)

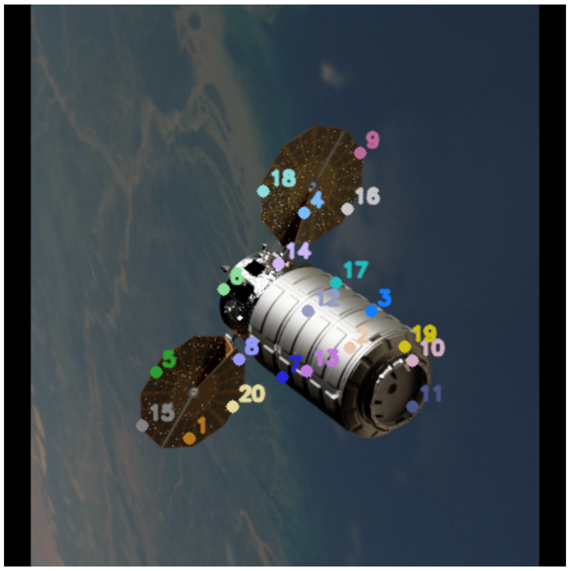


Figure 2. Synthetic Cygnus with 20 2D Keypoints

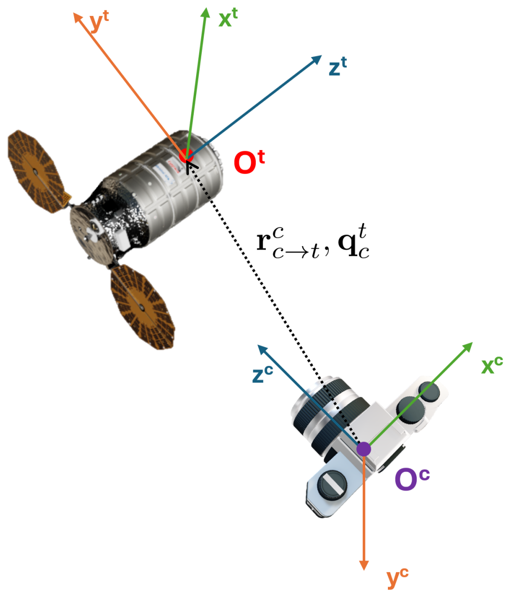


Figure 3. The Geometry of the Relative Pose Problem with Cygnus body-fixed frame, origin at O^t, and the Camera body-fixed frame, origin at O^c

ference the dropout layers are kept online and predictions are compiled for a desired amount of passes over the input data. The empirical mean and standard deviation of these samples can represent the model's epistemic uncertainty.[10]

In theory, MCD can be thought of as sampling the posterior distribution of the predictions $\mathbf{y}^*$, given the training data, $\mathbf{X}$ and $\mathbf{Y}$, and the new input, $\mathbf{x}^*$: $p\left(\mathbf{y}^* \mid \mathbf{x}^*, \mathbf{X}, \mathbf{Y}\right)$. To define this posterior, the posterior of the model's parameters (network weights and biases) $\boldsymbol{\omega}$ given the training data, $\mathbf{X}$ and $\mathbf{Y}$, $p(\boldsymbol{\omega} \mid \mathbf{X}, \mathbf{Y})$ should also be defined. Assuming the input training data are independent of the model parameters, $p(\mathbf{X} \mid \boldsymbol{\omega}) = p(\mathbf{X})$, applying Bayes Theorem and properties of conditional probability leads to

$$\begin{aligned} p(\boldsymbol{\omega} \mid \mathbf{X}, \mathbf{Y}) &= \frac{p(\mathbf{Y} \mid \mathbf{X}, \boldsymbol{\omega}) p(\mathbf{X} \mid \boldsymbol{\omega}) p(\boldsymbol{\omega})}{p(\mathbf{Y} \mid \mathbf{X}) p(\mathbf{X})} \\ &= \frac{p(\mathbf{Y} \mid \mathbf{X}, \boldsymbol{\omega}) p(\boldsymbol{\omega})}{p(\mathbf{Y} \mid \mathbf{X})} \end{aligned} \tag{1}$$

The marginal likelihood is

$$p(\mathbf{Y} \mid \mathbf{X}) = \int p(\mathbf{Y} \mid \mathbf{X}, \boldsymbol{\omega}) p(\boldsymbol{\omega}) d\boldsymbol{\omega} \tag{2}$$

With marginalization, properties of conditional probability, and assumptions on $\mathbf{y}^*$ and $\mathbf{x}^*$ the posterior distribution of $\mathbf{y}^*$ is

$$\begin{aligned} p(\mathbf{y}^* \mid \mathbf{x}^*, \mathbf{X}, \mathbf{Y}) &= \int p(\mathbf{y}^*, \boldsymbol{\omega} \mid \mathbf{x}^*, \mathbf{X}, \mathbf{Y}) d\boldsymbol{\omega} \\ &= \int p(\mathbf{y}^* \mid \boldsymbol{\omega}, \mathbf{x}^*, \mathbf{X}, \mathbf{Y}) p(\boldsymbol{\omega} \mid \mathbf{x}^*, \mathbf{X}, \mathbf{Y}) d\boldsymbol{\omega} \\ &= \int p(\mathbf{y}^* \mid \boldsymbol{\omega}, \mathbf{x}^*) p(\boldsymbol{\omega} \mid \mathbf{x}^*, \mathbf{X}, \mathbf{Y}) d\boldsymbol{\omega} \\ &= \int p(\mathbf{y}^* \mid \boldsymbol{\omega}, \mathbf{x}^*) p(\boldsymbol{\omega} \mid \mathbf{X}, \mathbf{Y}) d\boldsymbol{\omega} \end{aligned} \tag{3}$$

Model parameters are assumed to be independent of new data $\mathbf{x}^*$, and $\mathbf{y}^*$ is conditionally independent of $\mathbf{X}$ and $\mathbf{Y}$ because $\mathbf{x}^*$ and $\boldsymbol{\omega}$ are sufficient to characterize $\mathbf{y}^*$. The likelihood of the output given a particular model and input, $p(\mathbf{y}^* \mid \boldsymbol{\omega}, \mathbf{x}^*)$, is an aleatoric source of uncertainty because it accounts for noise in the data. The model parameter posterior distribution, $p(\boldsymbol{\omega} \mid \mathbf{X}, \mathbf{Y})$, is epistemic in nature because it characterizes uncertainty due to limited knowledge.[24] Substituting Eq. (1) and Eq. (2) into Eq. (3), the posterior prediction distribution is

$$p(\mathbf{y}^* \mid \mathbf{x}^*, \mathbf{X}, \mathbf{Y}) = \frac{\int p(\mathbf{y}^* \mid \boldsymbol{\omega}, \mathbf{x}^*) p(\mathbf{Y} \mid \mathbf{X}, \boldsymbol{\omega}) p(\boldsymbol{\omega}) d\boldsymbol{\omega}}{\int p(\mathbf{Y} \mid \mathbf{X}, \boldsymbol{\omega}) p(\boldsymbol{\omega}) d\boldsymbol{\omega}} \tag{4}$$

Resolving or approximating Eq. (4) is key to uncertainty quantification in neural networks. Its uncertainty is referred to as the predictive uncertainty.[9] Bayesian Neural Networks (BNNs) approximate Eq. (4) by treating model parameters as random variables. The parameters have some proposed prior $p(\boldsymbol{\omega})$, and the likelihood $p(\mathbf{Y} \mid \mathbf{X}, \boldsymbol{\omega})$ is determined given the data.[25] Training a BNN to learn $p(\boldsymbol{\omega} \mid \mathbf{X}, \mathbf{Y})$ is complex and often paired with variational inference methods, which seek to minimize the difference between an approximate distribution and $p(\boldsymbol{\omega} \mid \mathbf{X}, \mathbf{Y})$. Gal and Ghahramani[10] demonstrated an equivalence between training a BNN with variational inference and training a neural network with dropout. They approximated Eq. (4) with samples generated by inferencing with dropout activated (stochastic forward passes through the network) and Bayesian approximation of Eq. (4) when treating Eq. (1) as Gaussian processes. Their work provided a mathematical basis for epistemic uncertainty estimation of deep neural networks using MCD. The next paragraph will describe this process in terms of the presented equations.

Training with dropout in the MCD framework approximates $p(\boldsymbol{\omega} \mid \mathbf{X}, \mathbf{Y})$ from Eq. (1). Inferencing in MCD samples a set of weights from $p(\boldsymbol{\omega} \mid \mathbf{X}, \mathbf{Y})$, and passing a new input $\mathbf{x}^*$ through the network provides a sample of $p(\mathbf{y}^* \mid \boldsymbol{\omega}, \mathbf{x}^*)$. Repeating this process through multiple forward passes of the input effectively marginalizes the model parameters $\boldsymbol{\omega}$ from Eq. (3) and approximates the integral in this equation. Thus, the empirical mean and variance of the samples after multiple passes can be approximated as the mean and variance of the posterior prediction distribution $p(\mathbf{y}^* \mid \mathbf{x}^*, \mathbf{X}, \mathbf{Y})$.

Methodology. This work replaces the separate object detection and keypoint regression networks in Fig. 4 with `pytorch`'s Keypoint Region-based Convolutional Neural Network (Keypoint R-CNN)[26] to learn object and keypoint detection simultaneously. Keypoint R-CNN drops the mask prediction in Mask R-CNN,[27] a model for object detection and instance segmentation, and adds keypoint detection.

Fig. 5 displays the information flow in Keypoint R-CNN. Input images first encounter a feature extractor, which produces feature maps at different resolutions. Feature maps, the outputs of convolutional layers in a neural

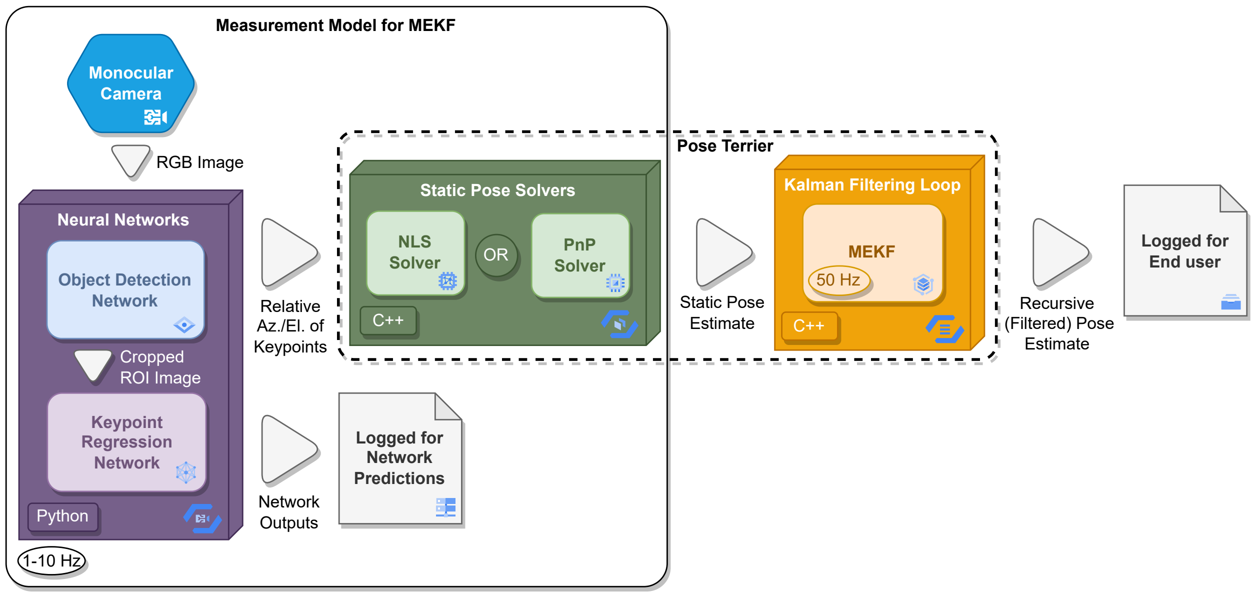


Figure 4. Vision Pose Estimation Pipeline Architecture

network, capture edges, textures, and other features in the data by applying filters known as kernels. These kernels slide across the input matrix, performing convolution operations (element-wise multiplications) on the overlapping regions to extract features. The resulting feature maps are handed to a Feature Pyramid Network (FPN)[28] to create rich multi-scale feature maps. In short, this process helps the model learn an object at different sizes. The feature extractor with the FPN are referred to as the model backbone. A Region Proposal Network (RPN)[29] takes the multi-scale feature maps and generates bounding boxes, also referred to as region proposals, that could contain objects. Both the proposals and the multi-scale feature maps are then fed into a Region of Interest Align (RoI Align) block. Here, the bounding boxes crop the feature maps. Proposal sizes typically dictate whether lower or higher resolution maps are cropped. For example, the RoI Align tends to select a lower resolution feature map for a larger region proposal and higher resolution feature maps for smaller region proposals. The cropped regions will be resampled to a fixed size to be fed into the bounding box and keypoint heads. Each head performs additional processing before handing off to a box predictor and keypoint predictor. The box predictor, which consists of two fully connected layers, classifies detected objects' classes and regresses their bounding box coordinates. The keypoint predictor, which consists of a transposed convolution layer that performs upsampling, regresses each detected object's keypoint locations. Detailed descriptions of Keypoint R-CNN components can be found in Ref. 30. Note, the model structure implemented in `pytorch` will not exactly mimic Fig. 5. For example, the bounding box and keypoints heads and predictors are nested inside the RoI Align head.

The deployability of Vision to low Size, Weight, Power, and Cost (SWaP-C) platforms for future operations necessitated several modifications: the feature extractor was changed from ResNet-50[31] to MobileNetv3,[32] the FPN was modified to adjust to the feature maps of MobileNetv3, and the fully connected layers in the bounding box head were replaced with depthwise separable convolutions.[33] The custom Keypoint R-CNN has 8.52 million trainable parameters and is 85% smaller than its standard implementation in `pytorch`.

As no dropout layers exist in the default Keypoint R-CNN, they are added to the end of each `Conv2dNormActivation` block in the feature extractor, FPN, RPN, bounding box head, and keypoint head; each of these five areas respectively has a single prescribed dropout probability and as whole they are denoted as $p = \{p_1, p_2, p_3, p_4, p_5\}$. Dropout layers are not added to the bounding box and keypoint predictors to avoid regularization in the output predictions of the network. Blocks with dropout are indicated with a diamond in Fig. 5. A total of 51 dropout layers are placed in Keypoint R-CNN, and their inclusion does not add trainable parameters to the network. The placement of dropout layers inside `Conv2dNormActivation` blocks is convenient and preserves regularization while maintaining meaningful feature extraction. `Conv2dNormActivation` is a composite block structure common in deep learning models that efficiently packages a 2D convolution layer that applies a convolutional filter to an input, a normalization layer that normalizes the output, and an activation layer that introduces non-linearity to the normalized output. The dropout layer probability distribution is the standard Bernoulli distribution with probability p, which is pictured in Fig. 6. In Fig. 6, if $q = .2$ and $x_n = 100$,

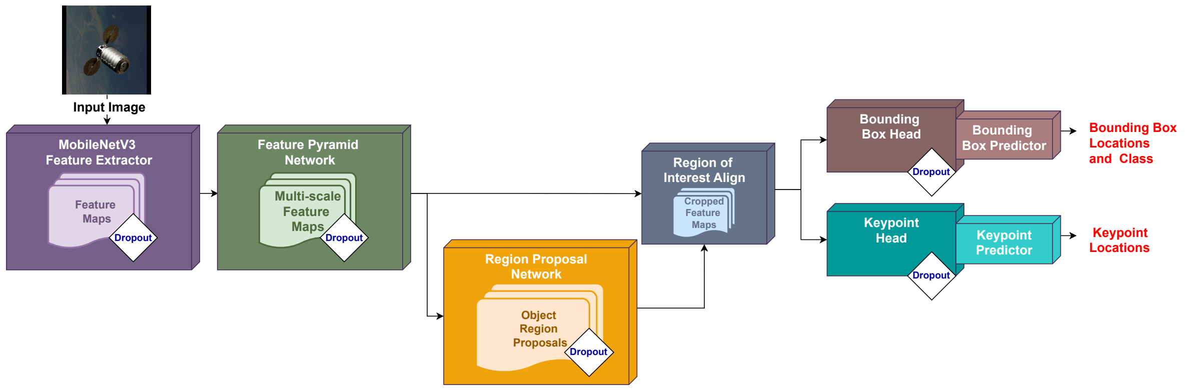
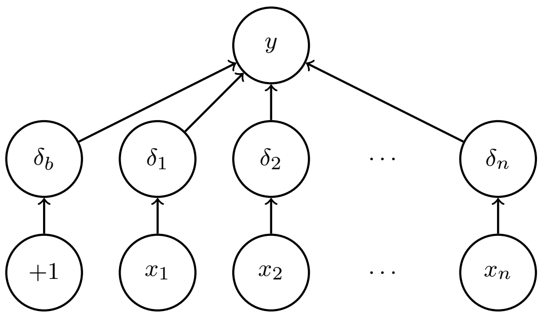


Figure 5. Modified Keypoint Region-based Convolutional Neural Network Architecture with Dropout Placement Indicated

one would expect approximately 20 input neurons to be zeroed out on any given pass of information from the input to the output layers. Dropout probability is a tuning parameter, and placing dropout layers within a network architecture often requires domain knowledge and experimentation. The randomness introduced by the dropout layers during training allows the model to update its parameters based on incomplete information, in effect injecting a controlled form of uncertainty into the model.



$$\delta_k = \begin{cases} 0, & q = 1 - p \\ 1, & p \end{cases}$$

Figure 6. Visualizing Dropout Layers with a Bernoulli Distribution, graphic adopted from Ref. 34.

In training, Keypoint R-CNN consumes the input images as a normalized tensor, the object classes, the truth bounding boxes, and the truth keypoints. In inference, Keypoint R-CNN consumes an input image and returns predicted bounding boxes, keypoints, and class labels along with measures of the model's relative confidence in its predictions. We train Keypoint R-CNN on 11,000 synthetic images of Cygnus generated in Blender, an open-source 3D computer graphics software, with varying backgrounds, orientations, sizes, and color gradients. There is a single Cygnus object in each image. Keypoint R-CNN learns to propose a Region of Interest (RoI) that may contain an object, classify the object, and simultaneously predict bounding box and keypoint coordinates within the RoI. A multi-task loss function that combines losses for object detection (classification and bounding box regression) and keypoint regression is deployed to allow the network to learn both object localization and keypoint prediction simultaneously. We exploit the `albumentations`[35] package to increase the domain randomization of the training data, by applying random transformations with some prescribed probability. These augmentations include brightness and contrast changes, the addition of sun flares, creating motion blur, and adding Gaussian noise.

The NLS in the Vision pipeline requires a transformation from keypoints in pixel coordinates to azimuth and elevation angles in radians. All training and inference imagery currently share the same camera sensor with a focal length of 50 millimeters (mm), image sensor size of 36 mm (width) x 24 mm (height) with square pixels, and an output image resolution of 512 x 512 pixels (height h by width w). The transformation involves shifting the keypoints from the image coordinate system (with the origin at the top-left corner) to the camera coordinate system (with the origin at the center of the image) and normalizing by the focal length in pixels, f_{px}. For n keypoints of the form $n_j(x, y)$ where $j \in \{1, 2 \ldots n\}$ and N signifies the entire set of keypoints, we have $N \in \mathbb{R}^{n \times 2}$ (e.g., $n = 20$, $N \in \mathbb{R}^{20 \times 2}$). The corresponding set in angle space is Θ. The transformation for the j^{th} keypoint from pixel coordinates (x_j, y_j) to azimuth (α_j) and elevation (β_j) angles is

$$\begin{aligned} \alpha_j &= \operatorname{arctan2}\left(\frac{x_j - \frac{1}{2}w}{f_{\text{px}}}, 1\right) \\ \beta_j &= \operatorname{arctan2}\left(\frac{y_j - \frac{1}{2}h}{f_{\text{px}}}, 1\right) \end{aligned} \tag{5}$$

The transformation of a variable $\mathbf{g}_N$, which contains n 2D keypoints in (x, y) pixel coordinates, to $\mathbf{g}_\Theta$, representing the corresponding set of n azimuth (α) and elevation (β)

angles, is represented by the F operator and given by

$$\begin{aligned}\mathbf{g}_\Theta &= F(\mathbf{g}_N)\\ &= \Bigg[\operatorname{arctan2}\left(\frac{\mathbf{g}_{N_{\{j,1\}}} - \frac{1}{2}w}{f_{\text{px}}}, 1\right),\\ &\quad \operatorname{arctan2}\left(\frac{\mathbf{g}_{N_{\{j,2\}}} - \frac{1}{2}h}{f_{\text{px}}}, 1\right)\Bigg]_{j=1}^{n}, \quad \mathbf{g}_\Theta \in \mathbb{R}^{n\times 2}\end{aligned} \tag{6}$$

The $\{j,1\}$ and $\{j,2\}$ notation indicates indexing into the first and second dimensions of $\mathbf{g}_N$, respectively. No uncertainties from the camera parameters are considered for subsequent MCD analyses.

Performance Metrics. This Monte Carlo Dropout study consists of selecting I images from an unseen test dataset, activating the model's dropout layers during inference, predicting the bounding box and the n keypoint pixel locations from the input images, and compiling these predictions for T trials. Cygnus' pose, scale, occlusion, and visual distortion, such as blur and glare, change from image to image along with the backgrounds. It should be noted that limiting the MCD trials to a single image may underestimate the uncertainty of the model because it will not account for a diversity of pose information in a target dataset. The model's uncertainty can be gauged from both prediction and error statistics.

Prediction statistics reflect the model's internal variability and are aggregated across the keypoints, trials, and images. The keypoint predictions, $\mathbf{y}_N^{*(i,t)}$, are tracked across the images and trials. The mean prediction for the keypoints, $\boldsymbol{\mu}_N$, is

$$\boldsymbol{\mu}_N = \frac{1}{I\cdot T}\sum_{i=1}^{I}\sum_{t=1}^{T}\mathbf{y}_N^{*(i,t)}, \quad \boldsymbol{\mu}_N \in \mathbb{R}^{n\times 2} \tag{7}$$

Eq. (7) is used in Eq. (8), but it does not offer significant insights for keypoint regression, notably when the target orientation is changing from image to image ($I > 1$). The prediction variance $\boldsymbol{\Sigma}$ quantifies the distribution or variability of the predictions and is a measure of the model's epistemic uncertainty. This spread is the uncertainty in the model weights, and it represents the model's confidence in its predictions given its training.[36] Viewing the prediction variance in this manner is straightforward when considering a single input with multiple forward passes ($I = 1$, $T > 1$). If predicting on out-of-distribution data such as real imagery of Cygnus, a high prediction standard deviation reflects the model's limited knowledge. The shorthand ($\sim$) is employed to denote that the preceding term in parentheses is repeated. The standard deviation at each keypoint is computed as

$$\boldsymbol{\sigma}_N = \sqrt{\frac{1}{I\cdot T}\sum_{i=1}^{I}\sum_{t=1}^{T}\left(\mathbf{y}_N^{*(i,t)} - \boldsymbol{\mu}_N\right)^{\circ 2}}, \quad \boldsymbol{\sigma}_N \in \mathbb{R}^{n\times 2} \tag{8}$$

Element by element operations are notated with the $\circ 2$ symbol. The spread in the predictions $\boldsymbol{\sigma}_N$ across the x and y dimensions is calculated as

$$\sigma_x = \frac{1}{n}\sum_{j=1}^{n}\boldsymbol{\sigma}_{N_{\{j,1\}}}, \quad \sigma_y = \frac{1}{n}\sum_{j=1}^{n}\boldsymbol{\sigma}_{N_{\{j,2\}}} \tag{9}$$

Reducing the standard deviation in this manner is necessary to populate Vision's static pose solver with a measurement noise matrix. The corresponding prediction quantities in the angle space are:

$$\boldsymbol{\mu}_\Theta = F(\boldsymbol{\mu}_N), \quad \boldsymbol{\mu}_\Theta \in \mathbb{R}^{n\times 2} \tag{10}$$

$$\boldsymbol{\sigma}_\Theta = F(\boldsymbol{\sigma}_N), \quad \boldsymbol{\sigma}_\Theta \in \mathbb{R}^{n\times 2} \tag{11}$$

$$\sigma_\alpha = F(\sigma_x), \quad \sigma_\beta = F(\sigma_y) \tag{12}$$

The measurement model for the Vision MEKF assumes additive noise modeled as a white Gaussian sequence: $\mathbf{z} = h(\mathbf{x}) + \boldsymbol{\epsilon}$ where $\mathbf{z}$ is the measurement, $h(\cdot)$ is the measurement mapping, $\mathbf{x}$ is the state, and $\boldsymbol{\epsilon}$ is noise.[22] The measurement covariance matrix of this measurement model is R. When deploying a dropout-infused Keypoint R-CNN in practice, its measurement error is not additive and more closely modeled by $h(\mathbf{x}, \boldsymbol{\epsilon})$ because the inference is deterministic since the dropout layers are turned off. This inconsistency will be handled in future work, and the quantified uncertainty is treated as additive here. Thus, the calculation of the first and second moments is assumed to be sufficient to model the Gaussian measurement noise. Furthermore, computing error metrics in our MCD analysis is possible because the inferencing imagery is synthetic and labeled.

We leverage the injection of uncertainty into Keypoint R-CNN from MCD to reframe the uncertainty quantification in terms of measurement noise. While many methods exist to estimate an unknown measurement covariance matrix within a filtering framework,[37] we implement a naive error analysis assuming the measurement noise is Gaussian and computing the error mean and covariance from a collection of measurements. This is an attempt to approximate the predictive uncertainty of Keypoint R-CNN, which as shown in Eq. (3) contains both aleatoric and epistemic uncertainties. Error for the i^{th} image and t^{th} trial at the N keypoint set is

$$\tilde{\mathbf{y}}_N^{(i,t)} = \mathbf{y}_N^{i} - \mathbf{y}_N^{*(i,t)} \tag{13}$$

The mean error across the images, trials, and keypoints estimates is

$$\tilde{\boldsymbol{\mu}}_N = \frac{1}{I\cdot T}\sum_{i=1}^{I}\sum_{t=1}^{T}\tilde{\mathbf{y}}_N^{(i,t)}, \quad \tilde{\boldsymbol{\mu}}_N \in \mathbb{R}^{n\times 2} \tag{14}$$

Eq. (14) is an estimate of the bias in the system. To conform to the white Gaussian noise assumption of the Vision filter, the bias is expected to be zero mean. Non-zero values at specific keypoints may indicate a lack of diversity in the training imagery or suggest improper learning by Keypoint R-CNN. The mean bias across the x and y dimensions is

$$\tilde{\mu}_x = \frac{1}{n}\sum_{j=1}^{n}\tilde{\boldsymbol{\mu}}_{N_{\{j,1\}}}, \quad \tilde{\mu}_y = \frac{1}{n}\sum_{j=1}^{n}\tilde{\boldsymbol{\mu}}_{N_{\{j,2\}}} \tag{15}$$

Error covariance is $\tilde{\boldsymbol{\Sigma}}$ and the corresponding standard deviation is

$$\tilde{\boldsymbol{\sigma}}_N = \sqrt{\frac{1}{I \cdot T} \sum_{i=1}^{I} \sum_{t=1}^{T} \left(\tilde{\mathbf{y}}_N^{(i,t)} - \tilde{\boldsymbol{\mu}}_N \right)^{\circ 2}} \ , \quad \tilde{\boldsymbol{\sigma}}_N \in \mathbb{R}^{n \times 2} \tag{16}$$

Eq. (16) is an estimate of the measurement noise standard deviation at each keypoint. With Eq. (16), visualizing a model's quantified uncertainty is possible by treating the standard deviation's first and second dimensions as the semi-major and semi-minor axes of an error ellipse centered about each keypoint prediction. This style of visualization is shown in Figs. 7 & 8 where the error ellipses are yellow and centered about each red prediction. Again, it is useful to reduce $\tilde{\boldsymbol{\sigma}}_N$ to scalar values:

$$\tilde{\sigma}_x = \frac{1}{n} \sum_{j=1}^{n} \tilde{\boldsymbol{\sigma}}_{N_{\{j,1\}}} \ , \quad \tilde{\sigma}_y = \frac{1}{n} \sum_{j=1}^{n} \tilde{\boldsymbol{\sigma}}_{N_{\{j,2\}}} \tag{17}$$

The measurement model covariance matrix can be defined with Eq. (17) as $R = \text{diag}\,[\tilde{\sigma}_x, \tilde{\sigma}_y]$. The corresponding angle space error statistics are:

$$\tilde{\boldsymbol{\mu}}_\Theta = F(\tilde{\boldsymbol{\mu}}_N) \ , \quad \tilde{\boldsymbol{\mu}}_\Theta \in \mathbb{R}^{n \times 2} \tag{18}$$

$$\tilde{\boldsymbol{\sigma}}_\Theta = F(\tilde{\boldsymbol{\sigma}}_N) \ , \quad \tilde{\boldsymbol{\sigma}}_\Theta \in \mathbb{R}^{n \times 2} \tag{19}$$

$$\tilde{\sigma}_\alpha = F(\tilde{\sigma}_x) \ , \quad \tilde{\sigma}_\beta = F(\tilde{\sigma}_y) \tag{20}$$

Root mean square error gauges overall prediction error for the dropout regularized Keypoint R-CNN in pixel coordinates:

$$\text{RMSE} = \sqrt{\frac{1}{n \cdot I \cdot T} \sum_{j=1}^{n} \sum_{i=1}^{I} \sum_{t=1}^{T} \left(\tilde{\mathbf{y}}_N^{(i,t)} \right)^\top (\sim)} \tag{21}$$

The angle space equivalent is

$$\text{RMSE}_\Theta = \sqrt{\frac{1}{n \cdot I \cdot T} \sum_{j=1}^{n} \sum_{i=1}^{I} \sum_{t=1}^{T} \left(F \left[\tilde{\mathbf{y}}_N^{(i,t)} \right] \right)^\top (\sim)} \tag{22}$$

Though not presented here, this uncertainty quantification framework can be extended to the bounding box regression outputs.

Results. Generally, MCD is performed on a single input with T forward passes, but quantifying measurement model noise requires a diverse set of poses and hence images. The 250 images used for the MCD study were taken from a dataset excluded from training that contains smooth relative pose trajectories. A subset of these trajectories are employed to assess the Vision filter performance after predictive uncertainty quantification. Monte Carlo Dropout is an input-specific method (see Eq. (3)): the uncertainty quantification is related to the specific input(s). Care should be taken to ensure the inputs for a Monte Carlo Dropout (MCD) study are representative of those expected during inference. In addition, MCD analyses may serve as a poor approximation of uncertainty when the inference imagery is considered significantly different from the training imagery (out-of-distribution data).

Four trained dropout configurations including the zero dropout baseline model are discussed in this section. Table 1 displays the prediction and error statics for the multiple image case of these models. For the zero dropout baseline, the predictions are deterministic and their variances, σ_x and σ_y, arise from the diversity of Cygnus orientations in the 250 images. Different orientations leads to different sets of keypoint locations for each image. Therefore, the prediction variance measures how the model's predictions vary across the images rather than model parameter uncertainty. This variability is an indirect measure of aleatoric uncertainty in the MCD imagery. The prediction standard deviations for $p = \{0, 0, 0, 0.025, 0.025\}$ and $p = \{0, 0, 0, 0, 0.05\}$ are marginally smaller or equivalent to the baseline, and the high dropout version, $p = \{0.1, 0.1, 0.1, 0.05, 0.05\}$ is marginally higher than the baseline. Such behavior implies the models may be robust to dropout-induced stochasticity with low amounts of epistemic uncertainty in the aggregate. Homogeneity in the 250 MCD images or insufficiently challenging training data may be responsible for these effects.

Table 1. MCD Prediction and Error Metrics in Pixels for Keypoint R-CNN Dropout configurations when $I = 250$ ***and*** $T = 300$

Dropout (p)	σ_x	σ_y	$\tilde{\sigma}_x$	$\tilde{\sigma}_y$	RMSE
$\{0, 0, 0, 0, 0\}$	61.7	90.8	16.6	17.0	20.8
$\{0, 0, 0, 0.025, 0.025\}$	61.5	90.7	9.4	9.6	11.5
$\{0, 0, 0, 0, 0.05\}$	61.4	90.8	11.2	13.7	15.1
$\{0.1, 0.1, 0.1, 0.05, 0.05\}$	61.9	91.6	24.5	25.1	28.6

Table 2 provides a clearer indication of how dropout affects model performance. Results in this table are for a single image with multiple stochastic passes. Increases in dropout lead to increases in the prediction variances, which indicate increased epistemic uncertainty. Dropout regularization is useful as it decreases the RMSE over the baseline for the $p = \{0, 0, 0, 0.025, 0.025\}$ and $p = \{0, 0, 0, 0, 0.05\}$ variants. Including dropout within the feature extractor and FPN layers had adverse effects on model accuracy and uncertainty. The prediction and error statistics for the $p = \{0.1, 0.1, 0.1, 0.05, 0.05\}$ model are higher than the other dropout configurations including the baseline. Dropout regularization is ineffective in the backbone layers because it can lead to the loss of important features, hindering a model's ability to learn data intricacies. Notice the prediction and error standard deviations are the same for a given dropout configuration in Table 2. For a single image, this behavior confirms that the observed prediction variance is entirely driven by epistemic uncertainty added via dropout. Dropout-induced epistemic uncertainty does not introduce systematic bias, resulting in a balanced spread around the truth keypoints. Consequently, the model's confidence in its predictions mirrors its true performance, indicating a well-calibrated model.

Table 2. MCD Prediction and Error Metrics in Pixels for Keypoint R-CNN Dropout configurations when $I = 1$ ***and*** $T = 300$

Dropout (p)	σ_x	σ_y	$\tilde{\sigma}_x$	$\tilde{\sigma}_y$	RMSE
$\{0, 0, 0, 0, 0\}$	0.0	0.0	0.0	0.0	16.6
$\{0, 0, 0, 0.025, 0.025\}$	3.1	0.8	3.1	0.8	6.8
$\{0, 0, 0, 0, 0.05\}$	4.4	1.8	4.4	1.8	7.8
$\{0.1, 0.1, 0.1, 0.05, 0.05\}$	14.7	8.0	14.7	8.0	27.1

For MCD analyses with many images, aleatoric uncertainty affects the results because the input data has variability. The comparable error standard deviations, $\tilde{\sigma}_x$ and $\tilde{\sigma}_y$, in Tables 1 & 2 are higher when the analysis has more imagery, suggesting MCD underestimates predictive uncertainty. The mean biases (see Eq. (15)) of the $p = \{0, 0, 0, 0.025, 0.025\}$ and $p = \{0, 0, 0, 0, 0.05\}$ models were less than a pixel, suggesting their CNN measurement model inputs are unbiased. The non-zero dropout models (e.g., $p = \{0.1, 0.1, 0.1, 0.05, 0.05\}$) had non-zero biases and its error uncertainties were higher than the the baseline.

The model trained only with a 5% dropout in the keypoint head, $p = \{0.00, 0.00, 0.00, 0.00, 0.05\}$, reached azimuth and elevation angle standard deviations close to one degree, see Table 3. This was near the default 1 degree measurement standard deviation for the MEKF before tuning. Limiting dropout to the keypoint head was more practical, and we're primarily interested in uncertainty quantification in keypoint regression over bounding box regression. Future exploration will investigate the interplay between dropout in the RPN, the bounding box head, and the keypoint head. The bias for the 5% configuration is zero mean in the azimuth and elevation space, and the uncertainty for a given keypoint location is around 12 pixels in either direction. Figs. 7 & 8 compare the predictive uncertainty estimates for the $p = \{0.1, 0.1, 0.1, 0.05, 0.05\}$ and $p = \{0, 0, 0, 0, 0.05\}$ models. The error ellipses in both figures are $\tilde{\boldsymbol{\sigma}}_N$, Eq. (16), of each model centered about their respective deterministic keypoint predictions. Larger ellipses imply higher predictive uncertainty for a given keypoint. Uncertainty in solar panel keypoint locations is more pronounced for the model with dropout in the backbone. As the solar panels are thin and have complex patterns and details (see Figs. 1 & 2), dropout in the backbone may lose important aspects of these features. Fig. 8 hints that the solar panel keypoints are difficult to learn because they exhibit higher uncertainties relative to other keypoints even when the model performs more accurate inference. The backbone dropout model ($p = \{0.1, 0.1, 0.1, 0.05, 0.05\}$) also has worse predictions because some green truth keypoints are not encompassed by their respective red open circle predictions.

Table 3. Error Statistics for a Model with Dropout $p = \{0, 0, 0, 0, 0.05\}$

$\tilde{\mu}_x$	$\tilde{\mu}_y$	$\tilde{\sigma}_x$	$\tilde{\sigma}_y$
0.18 px	0.41 px	11.16 px	13.71 px
$\tilde{\mu}_\alpha$	$\tilde{\mu}_\beta$	$\tilde{\sigma}_\alpha$	$\tilde{\sigma}_\beta$
0.01°	0.03°	0.89°	1.08°

Table 4 displays $\tilde{\boldsymbol{\sigma}}_N$ and $\tilde{\boldsymbol{\sigma}}_\Theta$, and it is helpful to look at predictive uncertainty across the keypoint locations to see which keypoints are noisy. Estimates of the 1st, 5th,

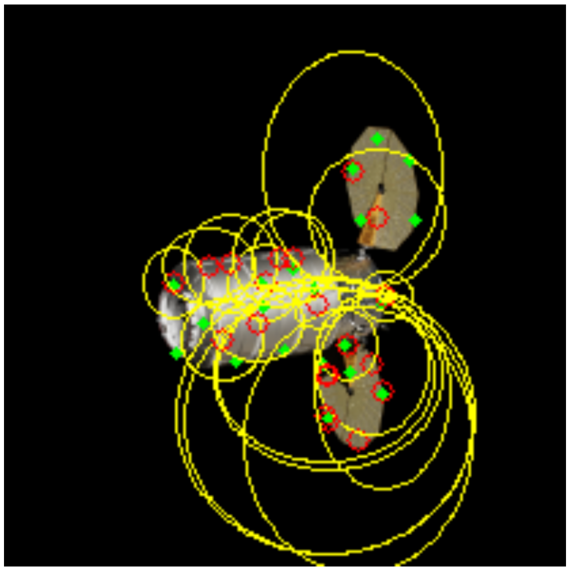


Figure 7. Projected Truth (Green) vs Inferenced (Red) Keypoints with Yellow Error Ellipses from $\tilde{\boldsymbol{\sigma}}_N$ ***for*** $p = \{0.1, 0.1, 0.1, 0.05, 0.05\}$

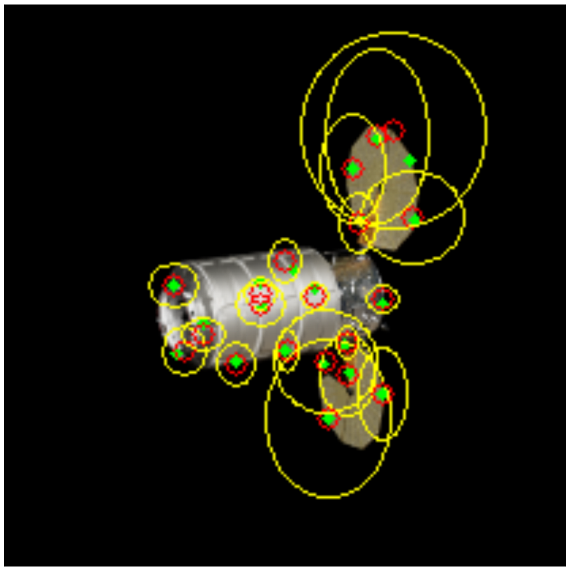


Figure 8. Projected Truth (Green) vs Inferenced (Red) Keypoints with Yellow Error Ellipses from $\tilde{\boldsymbol{\sigma}}_N$ ***for*** $p = \{0, 0, 0, 0, 0.05\}$

Table 4. Error Standard Deviations for the 20 Keypoints in Pixels and Degrees After $T = 300$ ***Passes and*** $I = 250$ ***Images (512 pixels x 512 pixels) with Dropout*** $p = \{0, 0, 0, 0, 0.05\}$

j	$\tilde{\boldsymbol{\sigma}}_{N_{j,1}}$	$\tilde{\boldsymbol{\sigma}}_{N_{j,2}}$	$\tilde{\boldsymbol{\sigma}}_{\Theta_{j,1}}$	$\tilde{\boldsymbol{\sigma}}_{\Theta_{j,2}}$
1	19.14	32.17	1.52	2.52
2	6.31	5.77	0.50	0.46
3	7.47	8.24	0.59	0.65
4	9.71	14.60	0.76	1.15
5	18.71	16.67	1.50	1.31
6	5.86	5.34	0.47	0.43
7	6.22	7.65	0.50	0.60
8	6.55	11.31	0.52	0.89
9	22.65	28.74	1.78	2.28
10	7.91	9.30	0.63	0.73
11	9.48	8.27	0.76	0.66
12	5.43	5.20	0.43	0.41
13	9.06	8.11	0.72	0.64
14	3.87	6.07	0.31	0.48
15	33.73	35.61	2.68	2.80
16	18.54	19.22	1.46	1.52
17	3.93	7.94	0.31	0.63
18	8.61	17.49	0.68	1.38
19	8.10	6.43	0.65	0.51
20	11.99	20.03	0.95	1.58
Means	σ_x	σ_y	σ_α	σ_β
-	11.16	13.71	0.89	1.08

9th, 15th, and 16th keypoints are more uncertain than the others. Results from the models in Table 1 reaffirm this pattern, signaling potential bias in the training data in terms of solar panel views, too little training time, and a need for more model parameters. The 1st, 5th, and 15th keypoints are on the left solar panel, and the 9th and 16th keypoints are on the right solar panel. Their locations are away from the cylindrical base of Cygnus, towards the edge of the solar panels. Ideally, when visualizing measurement model performance in the context of Figs. 7 & 8, the red hollow circles should encompass the green truth circles or contain the green truth circles within their error ellipses. This is the circumstance for the $p = \{0, 0, 0, 0, 0.05\}$ Keypoint R-CNN model, and the prediction for the 15th keypoint in Fig. 8 (red circle at the top of the figure) is not directly proximal to its truth location but its error ellipse covers it. Changing the MCD images still leads to similar levels of high uncertainty in the same keypoints, and future investigations of this non-uniformity and noise concentration in the solar panel keypoints is warranted.

Approximating the predictive uncertainty with the error metrics can help gauge network performance. A well-trained network should maintain consistent azimuth and elevation angle standard deviations across out-of-domain datasets where the trained model is expected to have reasonable ability to infer outside the training data (inductive bias). Later studies adding real images of Cygnus to the MCD analyses will directly address this idea. A statistically consistent filter with a quantified measurement noise can also indicate a well-trained Keypoint R-CNN model. After integrating the Keypoint R-CNN model with $p = \{0.00, 0.00, 0.00, 0.00, 0.05\}$ and its quantified uncertainty into the NLS, a parametric study using the trajectory image sets is undertaken to examine filter consistency. This is not a Monte Carlo simulation as the initial conditions nor truth are randomly perturbed; we have set aside distinct trajectories to undertake the study. Note, the attitude error is given by the Gibbs error quaternion, $\delta\mathbf{q_g} = \frac{\delta\mathbf{q_v}}{\delta q_0}$.

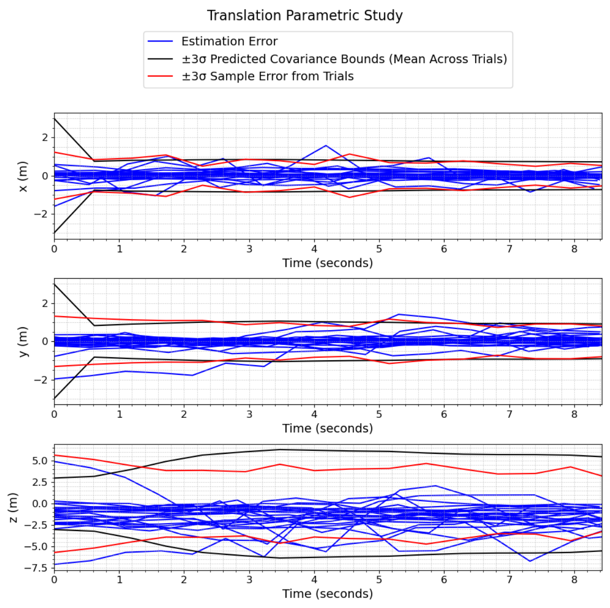


Figure 9. Parametric Study of Translation Error for 25 Image Trajectories

The parametric studies for both attitude and translation involve 25 trajectories and compare the filter's three sigma bound—representing its confidence in its estimates—with the parametric study's three sigma bound—an empirical measure of uncertainty based on the samples. The trajectories vary the initial pose, its propagation, and the space background. In Figs. 9 & 10 the filter tends to gain or maintain confidence over time, accepting most measurements from Keypoint R-CNN and the NLS. Notice that with limited trajectories the parametric study has error concentrated near the zero mean lines in both figures. Some measurements may be rejected due to large changes in attitude between the filter estimate and the measurement or large translation or attitude mean innovations relative to the filter's confidence bounds. Measurement rejection indicates a poor inference. The uncertainty bounds for the sample error in Fig. 9 track well with the x and y translation direction but less so with the z direction, which sees the filter becomes less confident over time. This may be explained by the difficulty in estimating depth information from 2D monocular imagery. In

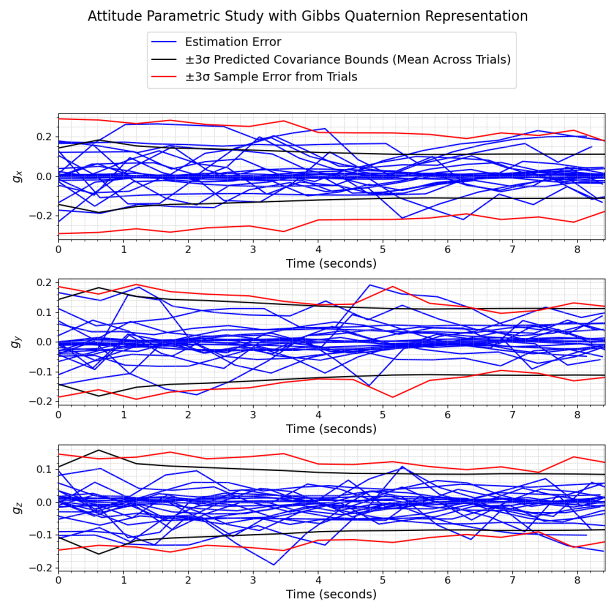


Figure 10. Parametric Study of Gibbs Quaternion Error for 25 Image Trajectories

the attitude example, Fig. 10 shows tighter mean filter three sigma bounds over the sample error bounds, suggesting that we could be underestimating the predictive uncertainty with our MCD method. The blue errors lines stray past the filter bounds at higher frequency than in translation, alluding to higher uncertainty in attitude estimation. Additional tuning with process noise may alleviate the underestimation of uncertainty, but there might not be enough trajectories to fully explore the parametric study. Overall, the filter appears consistent but additional research is needed.

Future Work. In future work, several areas will be reexamined, and new topics will be probed based on this work's findings. The navigation filter's assumption of a white Gaussian measurement noise and the non-additive nature of the regularized Keypoint R-CNN should be reconciled. Also, the prediction data used for MCD will be expanded to include real Cygnus imagery to investigate the aleatoric uncertainty that the model will encounter and SPEED+[38] imagery for comparison. Furthermore, adopting or developing a means to separately quantify the epistemic and aleatoric uncertainty in the system is crucial to further scrutinize our process of predictive uncertainty quantification. Our implementation of MCD for keypoint regression is tuning intensive[25] and exploring an analytical or numerical upper bound to the uncertainty may better inform the dropout design process. Adopting dropblocks as in Yelleni et al.[39] may warrant further consideration because these layers preserve spatial continuity. The dropblocks drop contiguous regions of feature maps instead of individual neurons. We will seek new methods to explore and limit uncertainty in the solar panel keypoints. Finally, a proper Monte Carlo study of this result will be pursued and involves defining a prior pose distribution determined from the pose distribution of both the training and (expected) inferencing imagery. This will require additional Blender development to dynamically generate the space of initial poses and their resulting trajectories.

Conclusion. We have applied the Monte Carlo Dropout method to inject uncertainty into a keypoint regression CNN that provides measurements to a pose estimation filter. A systematic approach to adding dropout layers to specific components of a model has been presented along with keypoint regression-centric equations for quantifying that uncertainty and extending the uncertainty quantification towards predictive uncertainty. A MCD analysis visualizing and evaluating the uncertainty has been showcased for trained Keypoint R-CNN models. Filter consistency analysis has been performed to demonstrate that the quantified uncertainty leads to a relatively consistent filter in the translation states but less so in the attitude states. Future work will directly consider aleatoric aspects of uncertainty, address restrictive assumptions, and develop additional capabilities.

References.

[1] C. P. Mark and S. Kamath, "Review of active space debris removal methods," *Space Policy*, vol. 47, pp. 194–206, 2019.

[2] N. T. Redd, "Bringing satellites back from the dead: Mission extension vehicles give defunct spacecraft a new lease on life - [news]," *IEEE Spectrum*, vol. 57, no. 8, pp. 6–7, 2020.

[3] D. Izzo, L. Pettazzi, and M. Ayre, "Mission concept for autonomous on-orbit assembly o...," in *56th International Astronautical Congress of the International Astronautical Federation, the International Academy of Astronautics, and the International Institute of Space Law*, 2005.

[4] L. Pauly, W. Rharbaoui, C. Shneider, A. Rathinam, V. Gaudillière, and D. Aouada, "A survey on deep learning-based monocular spacecraft pose estimation: Current state, limitations and prospects," *Acta Astronautica*, vol. 212, pp. 339–360, 2023.

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems* (F. Pereira, C. Burges, L. Bottou, and K. Weinberger, eds.), vol. 25, Curran Associates, Inc., 2012.

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

[7] T. H. Park, M. Märtens, M. Jawaid, Z. Wang, B. Chen, T.-J. Chin, D. Izzo, and S. D'Amico, "Satellite pose estimation competition 2021: Results and analyses," *Acta Astronautica*, vol. 204, pp. 640–665, 2023.

[8] R. C. Smith, *Uncertainty Quantification: Theory, Implementation, and Applications.* Philadelphia, PA: Society for Industrial and Applied Mathematics, 2013.

[9] Y. Gal *et al.*, *Uncertainty in deep learning.* PhD thesis, University of Cambridge, 2016.

[10] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *Proceedings of The 33rd International*

Conference on Machine Learning (M. F. Balcan and K. Q. Weinberger, eds.), vol. 48 of *Proceedings of Machine Learning Research*, (New York, New York, USA), pp. 1050–1059, PMLR, 20–22 Jun 2016.

[11] A. Kendall and R. Cipolla, "Modelling uncertainty in deep learning for camera relocalization," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4762–4769, 2016.

[12] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," in *Advances in Neural Information Processing Systems* (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), vol. 30, Curran Associates, Inc., 2017.

[13] A. Kendall and Y. Gal, "What uncertainties do we need in bayesian deep learning for computer vision?," in *Advances in Neural Information Processing Systems* (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), vol. 30, Curran Associates, Inc., 2017.

[14] L. Bramlage, M. Karg, and C. Curio, "Plausible uncertainties for human pose regression," in *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 15087–15096, 2023.

[15] L. Pasqualetto Cassinis, R. Fonod, E. Gill, I. Ahrns, and J. Gil-Fernández, "Evaluation of tightly- and loosely-coupled approaches in cnn-based pose estimation systems for uncooperative spacecraft," *Acta Astronautica*, vol. 182, pp. 189–202, 2021.

[16] K. Li, H. Zhang, and C. Hu, "Learning-based pose estimation of non-cooperative spacecrafts with uncertainty prediction," *Aerospace*, vol. 9, no. 10, 2022.

[17] B. Chen, J. Cao, A. Parra, and T. Chin, "Satellite pose estimation with deep landmark regression and nonlinear pose refinement," in *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pp. 2816–2824, 2019.

[18] T. H. Park, S. Sharma, and S. D'Amico, "Towards robust learning-based pose estimation of noncooperative spacecraft," in *2019 AAS/AIAA Astrodynamics Specialist Conference, Portland, Maine*, August 11-15 2019.

[19] K. Black, S. Shankar, D. Fonseka, J. Deutsch, A. Dhir, and M. R. Akella, "Real-time, flight-ready, non-cooperative spacecraft pose estimation using monocular imagery," in *31st AAS/AIAA Space Flight Mechanics Meeting*, AAS Paper 21-283, 2021.

[20] C. Yuksel, "Sample elimination for generating poisson disk sample sets," *Computer Graphics Forum*, vol. 34, no. 2, pp. 25–32, 2015.

[21] E. Lefferts, F. Markley, and M. Shuster, "Kalman filtering for spacecraft attitude estimation," *Journal of Guidance, Control, and Dynamics*, vol. 5, no. 5, pp. 417–429, 1982.

[22] S. Kaki, J. Deutsch, K. Black, A. Cura-Portillo, B. A. Jones, and M. R. Akella, "Real-time image-based relative pose estimation and filtering for spacecraft applications," *Journal of Aerospace Information Systems*, vol. 20, no. 6, pp. 290–307, 2023.

[23] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 56, pp. 1929–1958, 2014.

[24] S. Depeweg, J.-M. Hernandez-Lobato, F. Doshi-Velez, and S. Udluft, "Decomposition of uncertainty in Bayesian deep learning for efficient and risk-sensitive learning," in *Proceedings of the 35th International Conference on Machine Learning* (J. Dy and A. Krause, eds.), vol. 80 of *Proceedings of Machine Learning Research*, pp. 1184–1193, PMLR, 10–15 Jul 2018.

[25] B. Mostafa, R. Hassan, H. Mohammed, and M. Tawfik, "A review of variational inference for bayesian neural network," in *Artificial Intelligence and Industrial Applications* (T. Masrour, H. Ramchoun, T. Hajji, and M. Hosni, eds.), (Cham), pp. 231–243, Springer Nature Switzerland, 2023.

[26] S. Agarwal, K. Mierle, and T. C. S. Team, "Ceres Solver," 2022. Accessed: September 17, 2024.

[27] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[28] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[29] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 2017.

[30] H. Wang, L. Huang, K. Yu, T. Song, F. Yuan, H. Yang, and H. Zhang, "Camper's plane localization and head pose estimation based on multi-view rgbd sensors," *IEEE Access*, vol. 10, pp. 131722–131734, 2022.

[31] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[32] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, Q. V. Le, and H. Adam, "Searching for mobilenetv3," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[33] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1800–1807, 2017.

[34] E. Phaisangittisagul, "An analysis of the regularization between l2 and dropout in single hidden layer neural network," in *2016 7th International Conference on Intelligent Systems, Modelling and Simulation (ISMS)*, pp. 174–179, 2016.

[35] A. Buslaev, V. I. Iglovikov, E. Khvedchenya, A. Parinov, M. Druzhinin, and A. A. Kalinin, "Albumentations: Fast and flexible image augmentations," *Information*, vol. 11, no. 2, 2020.

[36] D. Feng, L. Rosenbaum, and K. Dietmayer, "Towards safe autonomous driving: Capture uncertainty in the deep neural network for lidar 3d vehicle detection," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pp. 3266–3273, 2018.

[37] J. Duník, M. Ŝimandl, and O. Straka, "Methods for estimating state and measurement noise covariance matrices: Aspects and comparison," *IFAC Proceedings Volumes*, vol. 42, no. 10, pp. 372–377, 2009. 15th IFAC Symposium on System Identification.

[38] M. Kisantal, S. Sharma, T. H. Park, D. Izzo, M. Märtens, and S. D'Amico, "Satellite pose estimation challenge: Dataset, competition design, and results," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 56, no. 5, pp. 4083–4098, 2020.

[39] S. H. Yelleni, D. Kumari, S. P.K., and K. M. C., “Monte carlo dropblock for modeling uncertainty in object detection,” *Pattern Recognition*, vol. 146, p. 110003, 2024.