Neural Computing and Applications
https://doi.org/10.1007/s00521-023-09094-8

ORIGINAL ARTICLE

An analysis of precision: occlusion and perspective geometry's role in 6D pose estimation

Jeffrey Choate[1] · Derek Worth[1] · Scott Nykl[1] · Clark Taylor[1] · Brett Borghetti[1] · Christine Schubert Kabban[2]

Received: 21 July 2023 / Accepted: 20 September 2023

Abstract
Achieving precise 6 degrees of freedom (6D) pose estimation of rigid objects from color images is a critical challenge with wide-ranging applications in robotics and close-contact aircraft operations. This study investigates key techniques in the application of YOLOv5 object detection convolutional neural network (CNN) for 6D pose localization of aircraft using only color imagery. Traditional object detection labeling methods suffer from inaccuracies due to perspective geometry and being limited to visible key points. This research demonstrates that with precise labeling, a CNN can predict object features with near-pixel accuracy, effectively learning the distinct appearance of the object due to perspective distortion with a pinhole camera. Additionally, we highlight the crucial role of knowledge about occluded features. Training the CNN with such knowledge slightly reduces pixel precision, but enables the prediction of 3 times more features, including those that are not initially visible, resulting in an overall better performing 6D system. Notably, we reveal that the data augmentation technique of *scale* can interfere with pixel precision when used during training. These findings are crucial for the entire system, which leverages the Solve Perspective-N-Point (Solve-PnP) algorithm, achieving 6D pose accuracy within 1° and 7 cm at distances ranging from 7.5 to 35 m from the camera. Moreover, this solution operates in real-time, achieving sub-10ms processing times on a desktop PC.

Keywords Real time 6D pose estimation · YOLOv5 object detection · Convolutional neural network · Perspective distortion · Synthetic imagery generation · Solve Perspective-N-Point algorithm

1 Introduction[1]

Accurate 6D pose estimation of rigid objects from color images is a paramount challenge with diverse applications spanning robotics and close-contact aircraft operations. This study explores the techniques to apply the YOLOv5 object detection CNN and Solve Perspective-N-Point (Solve-PnP) for the precise localization of aircraft 6D poses using color imagery. Conventional methods for object detection labeling encounter inaccuracies stemming from perspective geometry and are typically restrained to visible key points. This research demonstrated that with meticulous labeling, a CNN can achieve near-pixel accuracy in predicting object features, effectively grasping the distinct appearance of objects distorted by a camera's perspective. Additionally, the pivotal role of understanding occluded features in the training process is emphasized. While the inclusion of occluded features in the training slightly reduces the pixel precision, it leads to a substantial increase in the number of predicted features, including those initially hidden from view. Ultimately, this holistic approach results in an overall improvement in 6D pose estimation.

To underscore the necessity and significance of this approach, close-contact aircraft maneuvers, specifically aerial refueling is examined. The cost, risk, and necessity associated with aerial refueling have previously limited the adoption to military operations. However, if the process can be reliably and cheaply automated then there is the potential to democratize this technology by making it accessible beyond military circles. Thus, revealing the

Disclaimer The views expressed are those of the author and do not reflect the official policy or position of the US Air Force, Department of Defense, or US Government.

Extended author information available on the last page of the article

[1] See [5] for video demonstration of this work.

economic advantages highlighted by Nangia [29] and Parry [33], where it is shown that lightweight civilian aircraft could perform longer range flights with enhanced efficiency, thanks to the prospect of automated aerial refueling (AAR). Past attempts at AAR relied on differential Global Positioning System (GPS) and custom data-links to conduct their demonstrations [14, 44], thus making the cost untenable and failing to meet the military's need to operate in signal degraded environments.

Close-contact aircraft operations like aerial refueling require robust solutions in scenarios where human controllers, global positioning system data, and distance-based sensors may be unreliable or unavailable. Imagery-based sensors emerge as a potential solution, but their efficacy hinges on their ability to serve as both the primary source of information and be processed onboard. This paper identifies the necessary techniques for labeling and training YOLOv5, a CNN, to locate aircraft components with the precision required for the Solve-PnP algorithm. The objective is to achieve an aircraft's pose prediction accuracy within 7 cm and 1° of error. With these techniques in place, automated aircraft can extend their autonomous range, thereby reducing the need for human operators in high-risk missions and enhancing overall flight safety.

Beyond close-contact aircraft operations, this work addresses the fundamental challenge in the deployment of an artificial intelligence (AI) on autonomous aircraft - the need for airworthiness certification. The existing certification processes assume deterministic system behaviors with humans in the loop, and establishing a clear path for certifying algorithms designed for autonomous flight is currently lacking. [18]. Such a certification will be required to operationalize algorithms such as those designed in this article. Certifications prescribe the conditions under which algorithms can operate, such as proximity to population centers, times of day, aircraft compatibility, range between aircraft, in presence of precipitation, and many potential others. This process can be easier if a system's components are modular, understandable, individually testable, and individually certifiable. These are some of the reasons why object detection is used for the approach in this paper, instead of relying on a monolithic neural network that provides the 6D localization of an observed object. However, typical object detection still presents some issues for this application.

In typical object detection, a classified bounding box is drawn over the visible portion of an object in an image. This causes a mismatch with the bounding box's 2D center and the projection of the 3D geometric center of the item being observed, thus resulting in inaccurate 6D pose localization when using these mismatched bounding boxes. There are two reasons for this mismatch. First, the inherent nature of labeling visible portions of an object means the center point would not be guaranteed to match and hence training a neural network would lead to imprecise 6D localization. Second, perspective distortion introduces disparity between the 2D center of a bounding box and the 3D geometric center of an object when projected onto a 2D image plane, a concept elaborated in the Background section. The broader question that emerges is whether a CNN, such as YOLOv5, can predict bounding boxes that correct for perspective distortion and occlusions while maintaining low pixel error.

To show the necessity of both correcting for perspective distortion and training with the knowledge of occluded components, an ablation study was conducted. This study serves to underscore the importance of these techniques in achieving accurate 6D pose estimation. Additionally, the impact of various data augmentation parameters, particularly focusing on those that may hinder the network's ability to learn perspective distortion correction. There is an analysis in the Background regarding which parameters may interfere with learning perspective distortion, with detailed equations in Appendix 2. To further explore the intuition of what the neural network is learning, a final experiment took images and tiled them onto a larger image in different positions. This allowed for an accuracy and 2D error comparison between the various positions within an image as well as to the original image in order to explore how the CNN was learning to correct for perspective distortion.

In summary, this paper showcases the capability of the YOLOv5 CNN to effectively correct perspective distortion and predict occluded features. This investigation highlights the proper intuition for perspective distortion and demonstrates a method for correcting such distortion. This research shows the necessity for each of the steps in the process to attain an error of less than 1° and 7 cm at 7.5–35 m from the camera. This is significant because it demonstrates that the system is both accurate and fast enough to potentially be used on autonomous aircraft or for aerial refueling in the civilian sector. Additionally, this method can be applied to 6D localization of objects with occlusions, making it useful for robotics. It could also be applied for use with even more accurate and faster object detection neural networks which have yet to be invented. A video showing these findings is available at [5].

The contributions of the paper are:

1. A challenge to current best practice, augmenting scale during CNN training, via experiments showing degradation of pixel precision.
2. A novel set of mathematical operations, exploring how perspective geometry causes distortion on projection of 3D components to 2D bounding boxes.

3. Experimental results demonstrating that a CNN-SolvePnP system is performant when trained with knowledge of occluded components and equivariant to translation.
4. Experimental results showing ability of a CNN to learn distinct appearance caused by perspective geometry.
5. An application-specific trade-off where 6D performance is enhanced via an increase in true positives from occluded components, despite a slight decrease in 2D precision.

2 Background

2.1 Aerial refueling use case

Aerial refueling has been a crucial capability for military and civilian aircraft for over 90 years, enabling extended flight durations and acting as a true force multiplier. The history of aerial refueling dates back to the mid-1920s when the first successful aerial refueling was performed between two Airco DH-4B biplanes [27]. Past attempts in 2006 and 2015 with Defense Advanced Research Projects Agency, National Aeronautics and Space Administration, and Northrop Grumman have succeeded in demonstrations of AAR systems [14, 44], but these relied on expensive custom sensor and communication suites between the receiver and tanker aircraft.

The two main aerial refueling systems in use are the United States Air Force (USAF) boom and the Navy (also NATO) probe and drogue system. With the USAF system, stereo vision cameras are used by a boom operator to look backwards towards a receiver aircraft and move the boom into the proper position. The boom operator extends the boom into a receptacle for a receiver aircraft once the receiver is in the correct position. The Navy system employs a flexible hose that trails from tanker with a drogue, also referred to as a basket, behind it into which a receiver aircraft must guide its probe.

In order to operate in GPS denied environments, receivers and tankers need sensors and the ability to process the sensor data. Tankers already tend to have cameras facing backward, so if those cameras can be used, modifications to the aircraft could be minimal. Modifying receivers could be kept to a minimum if only upward facing cameras are needed and local processing could be used. Hence, the approach proposed here focuses on implementing 6D localization with only images and knowledge of the aircraft being observed; this paper focused on a tanker camera facing a receiver.

Inexpensively adding sensors and creating reliable enough autonomy could extend AAR to the civilian community [29, 33]. Parry and Hubbard focused on using a vision based approach exclusively for finding the refueling basket immediately before contact, while relying on GPS for all other aspects of the refueling process [33]. This is due to certification concerns with neural networks.

2.2 Center point correction - perspective geometry

One of the key features to this process is the proper training of the CNN by correcting for a mismatch in the 2D bounding box center and the projection of a component's 3D geometric center to the image plane. This mismatch occurs because the length and proportions between projected points on a flat image sensor are not invariant to projective transformations. This is often called projective distortion, but typically that term is not used to emphasize the effects of the image sensor being flat.

The causes of perspective distortion on a flat image sensor are due to three main factors: 1) the distance from the camera, 2) the angle that object is away from the principle axis, and 3) the rotation the object itself has. Perspective distortion ultimately occurs because a part of an object is closer to the camera than another part of the object. The amount of distortion is based on the proportion of the distance to the object versus the amount one part is closer than another part. The key nuance is learning the fact that the perspective changes the *proportion* of mismatch or distortion correction required based on these criteria, not just the number of pixels that an item may be distorted.

Each of these criteria inspire some general rules for perspective distortion and the mismatch it causes in the bounding box center points. First, the further away from the camera, the lower the proportion of distortion; the closer, the greater proportion of distortion, observed in Figs. 16 and 17 and Table 10. Second, the distortion is minimized to zero when the object itself is rotated such that it is parallel the image plane. Minimization also occurs when the angle of an object's rotation is complementary to the the angle created by the ray from the optical center to the object's center and the principle axis. The rotation effect has local maxima for the error at approximately the bisector of the two angles. This is stated as approximate due to the complex nature of the equation and the nonlinear effects of distortion; however, effects of rotation are still useful as a general rule to develop intuition. These effects are detailed via equations in Tables 12, 13, and 14 in Appendix 2.

The angle between an object and the principle axis has two general observations useful here. First, as a camera changes its orientation, the object's angle relative the image plane also changes, causing different distortions. This particular observation is difficult for humans to intuit because the retina is curved, thus when an individual changes their eye's orientation, the item does not change in appearance at all. The second observation for this angle changing is that it affects where the minimum and maximum values occur for the angle at which an object itself is rotated relative to the image plane. Additional examples, figures, equations, and tables of proportion changes are provided in Appendix 2 to assist readers with developing their own intuitions.

These behaviors of perspective distortion are critical to designing experiments to test how a CNN may learn to correct the distortion and how to best teach a neural network. For the aerial refueling use case, the observed object is typically similar in orientation to the camera, with only slight perturbations. It is possible that a network may learn to correlate the pixel position in an image to the amount of correction needed. If this is the learned method, then clipping an image could lose vital information about the position of the object in the image. Further examination revealed this was likely incorrect due to the amount of random rotation of the observed object in the testing data. Compared to a camera with a 56° field of view, a similar amount of distortion could occur for 25% of the image due to random rotation of the observed object. Thus, leading to another insight, perspective distortion leads to observed objects having a distinct appearance depending on their 3D location and orientation, hence the correction required could be calculated and learned based on this distinct appearance regardless of location in an image. This leads to the experiment in the Latent Space Perspective Distortion Experiment section testing the pixel precision of the network respective of the position in an image.

This analysis of perspective geometry and how it creates distortion also provides insights to the relationship between data augmentation parameters used during training and their potential effectiveness. Earlier it was noted that the distance an item is from the camera can affect the distortion. Hence, the size of an object could indicate the amount of correction required. Much of the existing research [3, 6, 8, 9, 12, 13, 15, 16, 28, 40, 45, 46, 53, 58, 59] on object detectors improve their results by varying the scale during training, which helps generalization across large data sets, but could also hinder pixel precision by confusing the neural network during training. There is also an apparent correlation to the position in the image, leading to a potential for mosaic and translation as data augmentation techniques to affect the results. Mosaic is a technique where multiple images are combined into a single training sample and translation involves shifting the object within the image to different positions. Both techniques are typically utilized to create more diverse training data. This lead to the experiment detailed in the Training Data Augmentation Optimization section testing networks trained with different training time data augmentation.

3 Related work

With many tankers having stereo vision on them already, a previous approach by Anderson et al., was to utilize these cameras for the 6D pose estimate. Here, the stereo block matching algorithm created 3D point clouds that would be compared to a truth point cloud via the iterative closest point method to gain a pose estimate [1, 2]. While this was able to meet the error threshold, stereo block matching is restricted to stereo vision systems, requires extremely precise calibration, and requires extensive filtering of noise and occlusions to be effective. Modern approaches have shown that it is possible to use neural networks with Solve-PnP on monocular color imagery to meet the error threshold required [26], the experiments here extend this work.

3.1 Key point vs object detection

Typical approaches which use Solve-PnP involve using key point finders with known 2D-3D correspondences. Many of the simpler key point finders like Scale-Invariant-Feature-Transform or Speeded-Up-Robust-Features will find points and create descriptions with their own methods, which can really only be matched to other 2D points found in similar images with those same methods, lacking a bridge to making 3D correspondences reliably. Some more modern approaches have tried to solve this blind Solve-PnP problem by using neural networks to create descriptors and create these correspondences [23, 43]. Without correspondences, the task of using Solve-PnP becomes an optimization problem to search for the best set of correspondences [42], which can be slow and unreliable.

Similarly, CNNs have been shown to be able to identify key points with their own descriptors that are more resilient to image changes than the legacy methods and have extremely low latency[10, 11, 21, 25]. However, all of these key point identification methods thus far hide information inside their respective algorithms with custom descriptors for features that are found, which cause difficulty if used for 2D-3D correspondences. These custom descriptors are less explainable and suboptimal for usage with 6D localization. Consequently, generic key point finders and descriptors were not employed.

Other CNNs solve the correspondence issue by using landmark detection [15, 24, 54] and object detection. Often landmark detection is also called key point detection without distinction to the earlier methods. Landmark and object detectors are able to both classify and provide 2D localization of individual features. In landmark detection, a neural network is trained to find pixel locations of generically classified features in an image. An example of this is a person's knee or nose. At the time of writing, landmark detectors did not exhibit comparable execution speed, scalability, or the level of adoption achieved by leading object detectors. Nevertheless, landmark detectors present a promising alternative to the object detection for 6D localization, and recent advancements in object detection CNN design could certainly be leveraged for landmark detection.

You Only Look Once, YOLO [37] is a family of algorithms that was originally released at the Conference on Computer Vision and Pattern Recognition in 2016 from Joseph Redmon. Redmon released further versions [38, 39] the next two years following after which the algorithm was continued by others in the computer vision community, with many more iterations addressing modernizations in neural networks such as skip connections, cross-stage partial connections, inception modules, and different training techniques.

In general, the YOLO family of CNNs has a pre-trained classification network as its backbone. The neck, middle layers, of the network utilizes a feature pyramid, spatial pyramid pooling, and path aggregation networks to identify features at various scales as well as feed multiple output convolutional nodes to produce multiple bounding boxes for a given image. After the backbone is pre-trained, the neck is attached and the entire network is further trained on imagery which has bounding boxes labeled. For the work in this paper, YOLOv5 [19, 57] was chosen. This was due to its ease of adaptability, flexibility in precision vs speed, and precision achievable at speed.

3.2 Other pose detectors

In recent years, there has been a surge in other approaches that use CNNs for 6D pose localization. Kehl [20] developed a single shot 6D pose estimator which works in two stages: first, creating bounding boxes around objects to clip them into their own images at a specific size, then running other neural networks against that, which are essentially trained as classifiers based on the viewpoint. Xiang [52] developed a single neural network that is able to output the segmentation, translation, and orientation of objects by being trained on that data.

Pix2Pose [31] has an innovative approach, they apply a custom coloring of each object in an image where that coloring is associated with the 3D coordinate of the skin of the object, hence creating unique correspondences between color and positions on that object. They then train networks to create images with that coloring, then send those correspondences to Solve-PnP to estimate the 6D localization. Finally another neural network, BB8 [36], is trained to identify 3D bounding boxes for items, overlay them onto an image, and use the 8 corners of the resulting 3D bounding box as inputs to Solve-PnP to get a final 6D localization. [4, 7, 32, 34, 35, 47, 55, 56] are very similar to these approach with slight variations, all basically doing direct pose regression from a CNN. There are many other examples which use combinations of these approaches and add in depth data [17, 22, 41, 48–51], however depth data is outside the scope of the desired solution here.

Each of these methods show great promise using neural networks, but suffer from a similar issue: the concealing of the learned features inside of them, making them difficult for humans to understand. This could lead to difficulty certifying them or testing them. The difficulty is increased for monolithic systems like these, hence a lack of modularity. This increase in difficulty is because if a change occurs then the entire system would need to be re-tested and re-certified, as opposed to a single module. These are some of the reasons our approach utilizes an object detector with Solve-PnP, as they can be made small and fast with the ability to identify components in human readable ways and each step of the process can be tested independently and as a whole.

4 Methodology

4.1 Overall process

In order to conduct an ablation study on the importance of each step in the system, one must first understand the overall system designed. The system utilizes a trained CNN, YOLOv5, to draw bounding boxes around aircraft components in an image. The CNN provides a classification used later for 2D-3D correspondences and the 2D center point of the bounding boxes. If at least four components are found, then those correspondences and center points are utilized by Solve-PnP to calculate a pose for the aircraft. The ablation study is conducted by training multiple neural networks from the same set of images; different labeling techniques are used to train the multiple neural networks.

In the ablation study, the labeling techniques can be grouped into two main categories: visibility and perspective correction. For visibility, neural networks are trained from labels created either from only visible points or both visible and occluded points using "x-ray" vision, which was conceptually created via simulation and perfect truth data. For perspective correction, neural networks are trained either from labels created from data which corrects the perspective distortion in the bounding box centers or from data which does not. The metrics used to compare these neural networks are the average center point error (in pixels) for each bounding box in an image, the true and false positives a network identifies, and the position and orientation of the 3D model prediction.

4.2 Data generation

To train a CNN and assess the system, labeled imagery must be generated. This data consists of images, a label truth file for each image, and a pose truth file for each data set. This data is generated with AftrBurner [1, 26, 30], a 3D graphics engine maintained by a team at the Air Force Institute of Technology. AftrBurner imports a ".stl" file defining an aircraft model, renders that model, imports point clouds defining 3D components of the previously imported model file, saves rendered scenes as images, and saves the associated truth files. All aircraft model files used were publicly available files purchased and customized for this application; no official Air Force information or files were utilized. The images are sized at 640x480 pixels, with 56° horizontal field of view, and a perfect camera calibration is assumed.

To create 2D bounding boxes, the 3D model required a file defining the components on the aircraft. This was accomplished by creating point clouds on the aircraft by identifying the points on the skin of the 3D model in the model's coordinate system. Each of these point clouds is named accordingly with the component they resemble on the aircraft; 50 were created. These 3D model points defining the components are projected to the image in order to create bounding boxes for each component. At this point, occlusion checks are able to be made or omitted. Additionally, the perspective distortion which causes the center point from the 2D bounding box to misalign with the 3D component's geometric center point projection can be corrected during this process.

See Figs. 1 and 2 for an example point cloud of dots, with the red, or off-color, dot indicating the 3D component's correct center point projection on the image and a small green crosshair indicating the uncorrected bounding box center. Figure 3 includes an example of expanding the bounding box to correct for the perspective distortion. This example had approximately 9 pixels between misaligned and corrected bounding box centers.

The occlusion data allow filtering of point clouds to control generating bounding boxes based on either the visible portion or the entire component. At least 25% of a component's points must be visible to consider it visible and its center point must be visible. The process for the center point correction utilized in this study was to project the 3D geometric center point, $True_x$ and $True_y$, to the screen then expand the bounding boxes via taking the new correct center point and expanding the width and height. The process is shown in Algorithm 1

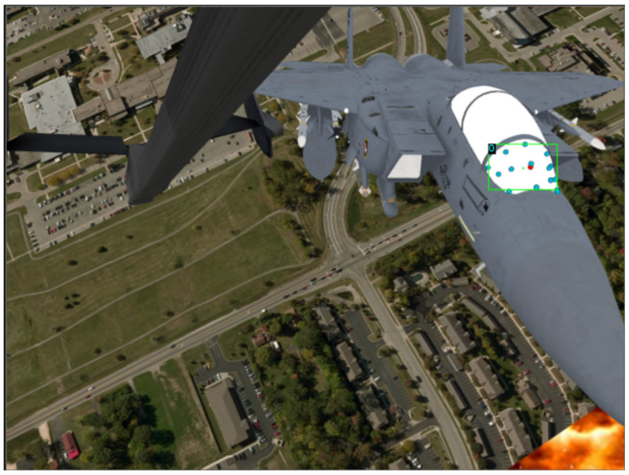

Fig. 1 Non-corrected bounding box and component point cloud

Algorithm 1 Center Point Correction Process

```
1. $\delta_x = |True_x - BBox_x|$
2. $\delta_y = |True_y - BBox_y|$
3. $BBox_x = True_x$
4. $BBox_y = True_y$
5. $Width_{new} = Width_{old} + 2\delta_x$
6. $Height_{new} = Height_{old} + 2\delta_y$
```

The result of these labeling strategies is 4 label files being generated for each image, with the resulting trained networks being named with the following convention:

1. Vs_CC - only visible points with center point correction
2. Vs_Nc - only visible points with NO center point correction
3. Xr_CC - All points regardless of occlusions with center point correction
4. Xr_Nc[2] - All points regardless of occlusions with NO center point correction

To robustly train the neural network, various attributes were randomized within the AftrBurner engine to generate the data set. The orientation of the receiving aircraft was randomly rotated up to 5° in the roll, pitch, and yaw. The position was randomly set between 7.5 and 35 m from the tanker camera in such a way to ensure the center of the aircraft was in the viewing frustum of the camera. Images without aircraft were used 5% of the time. The background

[2] The abbreviation "XR" was inspired by the concept of "x-ray" vision, which was conceptually created via simulation and perfect truth data.

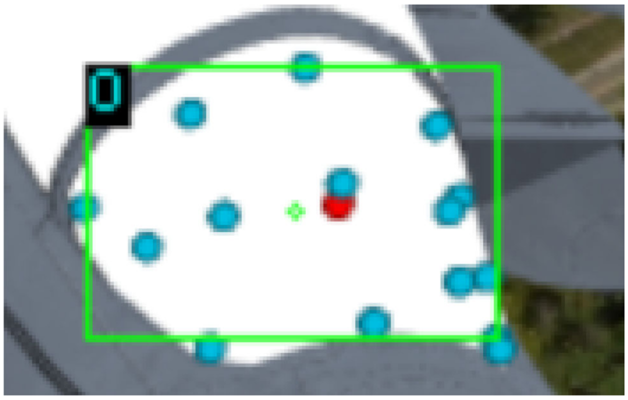

Fig. 2 Example bounding box without correction

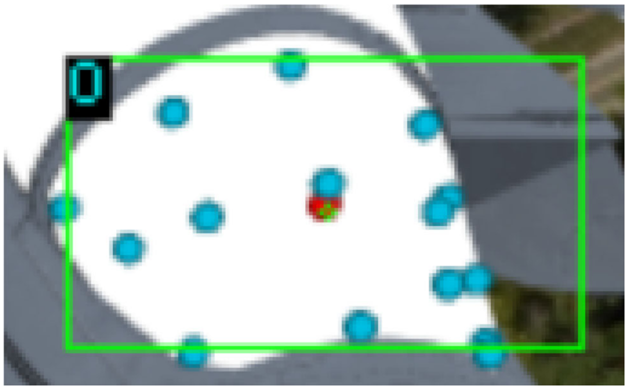

Fig. 3 Example bounding box with correction

sky box, orientation of the camera versus the ground, and lighting positioning was also randomly set for each image. Finally, the position of the refueling boom was randomly set in each image, causing potential occlusions to the aircraft in the image. Examples of these images are seen in Fig. 4.

4.3 Neural network training

The main experiment utilized the randomization of imagery attributes mentioned in the Data Generation section, 40,000 images, and the associated truth data. The first 30,000 images were used for training, and a further 5000 for validation during training to track and utilize early stopping criteria of 20 epochs. The last 5000 images were used for testing and comparing each of the trained models across the metrics mentioned in the Introduction section. When training, 1000 epochs was chosen as a top limit for the number of epochs to be used, with 20 epochs as the patience criteria for early stopping.

In addition to the data augmentation detailed in the Data Generation section, training time data augmentation was also utilized. This includes randomizing up to 70% saturation, 1.5% hue, 40% value, 3° rotation, 10% mixup, 10% translation, mosaic, and 90% scale for each training image. This is according to recommended defaults by the YOLOv5 team [19], with the exception of left-right flipping of the image; this was excluded because the object being detected is nearly left-right symmetrical, with few identifying symbols on each side and the components being predicted are respective of the side of the aircraft. An additional three networks were trained with different hyperparameters for training in order to explore the best method of teaching perspective distortion of a 3D object within an image, further detailed in Sect. 4.5. Each of these additional networks were trained with labeling of all components in the image regardless of occlusions and with center point correction enabled.

The YOLOv5 team has various pre-trained models available for transfer learning, with larger models being more accurate, but slower. The YOLOv5 small model, YOLOv5s, was chosen for transfer learning for this project due to the precision, accuracy, and speed it was able to provide. Other relevant parameters used for training were a batch size of 32, 0.0005 weight decay, a learning rate of 0.01, and 0.937 momentum. The two visible networks stopped from the early stopping criteria while the other Xr_CC and Xr_Nc networks trained for the full 1000 epochs; details of all networks trained are in Fig. 8.

4.4 Neural network execution

The object detection neural network is able to utilize various parameters to choose the precision versus recall needed, maximizing true positives and minimizing false positives. The main parameter to be modified is the objectiveness confidence. The confidence factor chosen for the experiments here was 0.85 because that was the maximum before a precipitous drop in F1 score by many of the networks used. This chosen value allowed no false

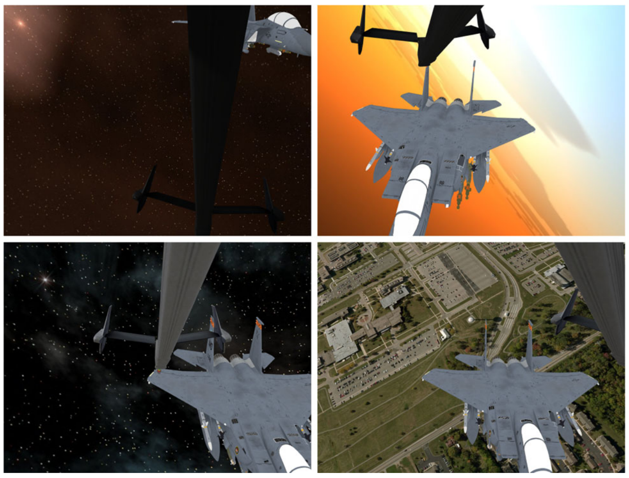

Fig. 4 Variability in training data for CNN: synthetic image examples

positives when there was no aircraft in the image and allowed adequate correspondences for Solve-PnP. Further fine tuning of the confidence could be beneficial for a future deployed system.

In this experiment it is assumed that only one of each component may appear in each image, allowing us to choose the largest confidence result for each component as the method to remove duplicates. In a deployed system, techniques like masking, clipping, or segmentation may be required in concert with the system proposed here to remove observations from other aircraft in an image.

4.5 Data augmentation training hyperparameter optimization

To address perspective distortion, an experiment was designed to explore which data augmentation to use during training because the typical data augmentation used for training CNNs for object detection utilize methods that could make learning perspective distortion difficult.

Some of the potentially troublesome techniques are mosaic, translation, and scale. Mosaic takes patches of four images and stitches them into one, essentially making the position of components in the image change. Translation also moves components within an image. These two techniques will matter greatly if the network is learning based on the position in the image, but will not if the network is learning based on the distinct appearance of the item as the indicator of perspective distortion. In addition, the distance an object is away from the camera changes the proportional amount of distortion, therefore scaling the image directly affects the perspective distortion.

The analysis of how perspective affects the appearance of objects in imagery led to the training of an additional three networks. The Xr_CC network mentioned previously is trained with complete knowledge of occluded components, center point correction, and with the data augmentation hyperparameters as outlined above. Each of the other networks were also trained with complete knowledge of occluded components and center point correction enabled. The Xr_CC_nS was trained like the Xr_CC but without scale augmentation; nM_nT was trained without mosaic and without translation; and nM_nT_nS was trained without mosaic, translation, or scale. These four models were used to assess the best technique with which to train in light of perspective distortion requiring correction.

1. Xr_CC - Default hyperparameters, no left to right flip
2. Xr_CC_nS - Same hyperparameters as Xr_CC, no modification to scale
3. nM_nT - Same hyperparameters as Xr_CC, no mosaic or translation
4. nM_nT_nS - Same hyperparameters as Xr_CC, no mosaic, translation, or scale

4.6 Latent space perspective distortion investigation

To further explore how the neural network was learning to correct for perspective distortion in the center points, another experiment was designed. To correct the position of a 3D object's center point, a neural network could learn based on two potential measures: the 2D location in the image, and/or the distinct appearance and proportions of the object due to the distance and orientation from the camera.

To assess the impact of position in the image, test images were superimposed onto a larger image of the virtual sky for analysis. This test took 520 images from the test data set of 640x480 sized images and superimposed them onto nine different positions of a 4096x3072 sized image, creating 9 more sets of images for testing. These positions included the combinations of top, bottom, middle, left, and right; a notional example is in Fig. 5. No black boxes are used in actual test images, rather there would be sky in those locations; it is only used here to show possible positions. Then inference was run with the Xr_CC_nS model on each set of images for comparison; this model is defined in the hyperparameter study. The comparison conducted here was for the average center point error and the true/false positives; the 3D error was not assessed as part of this comparison.

4.7 Solve-PnP error characterization

The usage of a CNN to detect individual components of an aircraft requires truth label files to be created, which allows for error characterization of the truth data itself. The truth label data for each of the labeling techniques was utilized as inputs to the Solve-PnP algorithm to characterize Solve-PnP and the system. The intent of this was to show the behavior of Solve-PnP across the ablation study. This allowed the limiting of error analysis to images for which the truth data would allow solutions. The Error from Truth Labels section of the results shows the 3D error when utilizing the truth labels and the number of images with valid results for further comparison.

5 Results and discussion

This section is composed of four subsections detailing the results of the various experiments and discussing what those results could indicate. The first is an examination of the truth values from the different labeling techniques and how Solve-PnP performs given those perfect truth values. The next section is an analysis of the ablation experiment with results from entire system using each of the different labeling techniques. Then details on the experiment where different training time data augmentation parameters were used and which performs the best with regard to perspective distortion is provided. Finally, there is a discussion on the experiment designed to test if the neural network was learning to correct for perspective distortion through the position in the image or the distinctive appearance of object. This last experiment led to a potential cause of false positives that is also discussed at the end of the section.

5.1 Error from truth labels

In order to evaluate the error of the overall system, the results of each labeling scheme using their truth labels was analyzed. These truth labels were used to directly make Solve-PnP predictions and limit the further analysis to that of the images in which solutions are possible. This was useful in characterizing the potential of the system. Figures 6 and 7 show each of the labeling techniques and their associated errors for position and orientation magnitudes of error. Each plot contains a dashed line representing the error threshold for the system of 7 cm and 1°, respectively. Table 1 explicitly presents the mean and standard deviation error of the Xr_CC labeling technique with center point correction.

The Vs_CC labeling technique exhibited outliers in rotation and position, which can be attributed to the sensitivity or fragility of the Solve-PnP algorithm when given only 4 points. To address this, a threshold of 6 components was established and is used in Fig. 8 to distinguish between good and poor amounts of components. Additionally, when Solve-PnP utilizes 6 points, it allows for direct linear transformations, enabling an estimate of the camera's intrinsic matrix and further refinement of the results. Importantly, when there are less than 6 components found in an image, the system does still make a prediction and the various figures and metrics all include such results.

The label data analysis revealed that predictions were possible for 4729 of the images. This occurred because, intentionally, 5% of the images in both the training and test data sets were generated without any aircraft, aligning with best practices. Consequently, these images lacked any aircraft presence. Interestingly, none of the trained models generated false positives when using a confidence threshold of 0.85 for the images without aircraft.

The XR_CC labeling strategy proved effective in providing sufficient truth labels, enabling the application of the Solve-PnP algorithm to make predictions for all 4729 images when provided truth data. In contrast, when using a labeling technique restricted to visible components, predictions were possible for 4586 images (96.98% of predictable images) in the case of images with center point correction and 4715 images (99.70% of predictable images) for images without center point correction. The difference in performance between these two visible labeling techniques can be attributed to the center point correction technique's requirement for the center point to be visible, resulting in slightly fewer visible points compared to the data without center point correction.

This analysis yields two interesting concepts. First is that a neural network trained on data without center point correction is unlikely to reliably predict under the desired error threshold. The next item is that the position error through the system is potentially 8% of the threshold in the best case, observed through Table 1. This exemplifies the difficulty of the problem space. This error is likely due to compounding of numerical precision issues between the simulation, various truth files, and limits on floating point variables respective of the scale at which the simulation

Fig. 5 Notional example superimposed images

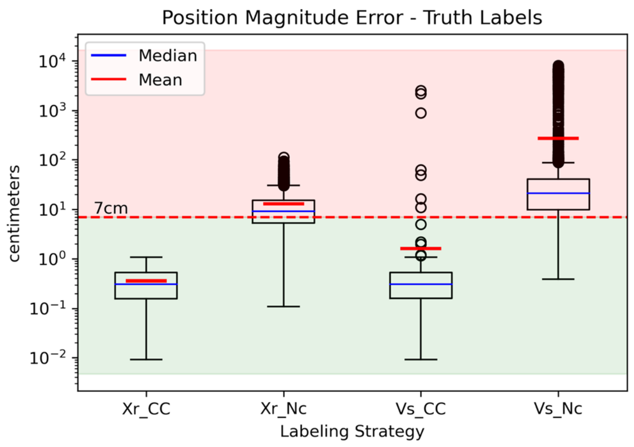

Fig. 6 Solve-PnP position error from truth labels

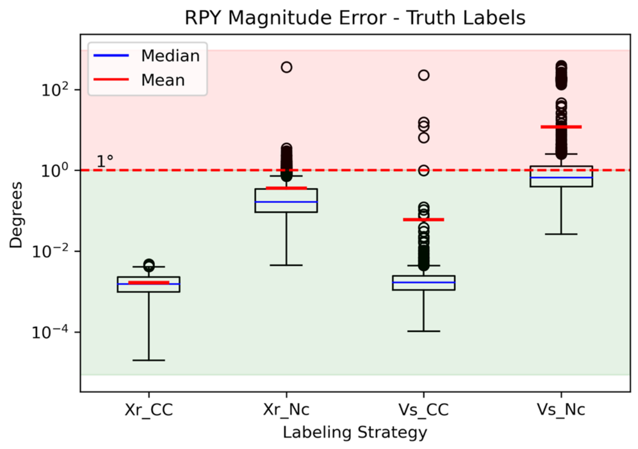

Fig. 7 Solve-PnP rotation error from truth labels

Table 1 Error from best truth data

Mean	0.36 cm	0.0017°
Standard deviation	0.23 cm	0.0009°

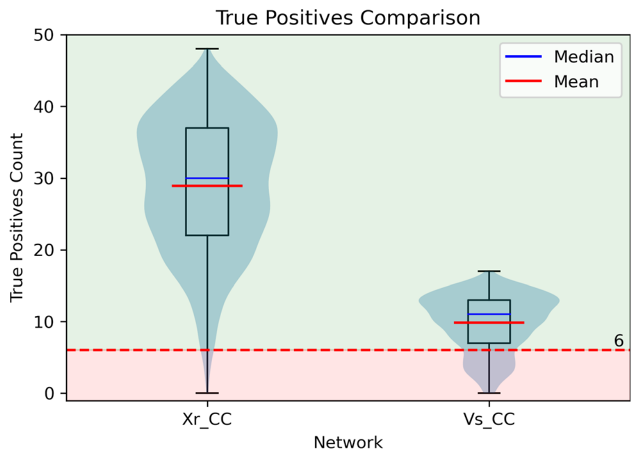

Fig. 8 Visibility comparison of true positives

was running. It is left to future work to optimize this aspect of the system.

The next interesting observation is that the orientation predictions were fairly accurate, even for the data sets without center point correction. This is likely due to the nature of Solve-PnP with RANSAC minimizing the error from the data it is sent. Additionally, when only provided 4 features Solve-PnP cannot use multiple samples to reduce the error, hence is more likely to produce a spurious output even when provided very good data. This is an indication that the Solve-PnP part of the pipeline is robust to some error, but it is vital to provide adequate amounts of data in order to allow it to perform the processes which make it robust. For Solve-PnP, when provided 6 or more points it can potentially even correct for intrinsic errors of the camera.

Overall, this initial analysis shows that the labeling scheme of center point correction is able to meet the error thresholds. While also showing that not using center point correction causes the error thresholds to typically be exceeded by Solve-PnP. Thus, a system designed with this approach should have sufficiently low 6D error if able to predict these labels with sufficient precision.

5.2 Labeling strategy ablation study

This subsection details the results of the ablation study, examining the effects of training a network with knowledge of occluded components or not, and correcting for perspective distortion in the centers of the bounding boxes or not. First the performance of models which were trained with and without knowledge of occluded features are compared. Then, the ability of the model to learn perspective distortion is shown via discussion of the amount of error and amount of correction required.

5.2.1 Training without regard to occlusions

The first comparison is between the models trained with knowledge of occluded components and trained without that knowledge. For this part of the analysis, the center point corrected models were used. Figure 8 shows box and whisker plots comparing the two networks. The dotted line indicates 6 points being detected. That is a significant number because it is the threshold for Solve-PnP to utilize direct linear transformations and conduct iterations to reduce error from noise in the measurements.

Figure 8 reveals the labeling technique has a large effect on the number of features that can possibly be observed. The technique used with XR_CC found nearly three times as many components on average. This is significant because it allows the Solve-PnP algorithm more opportunity to reduce error and potentially even correct for camera calibration issues. This is also an indication of the difficult nature of labeling components on a 3D object. Labeling 3-dimensional components situated on an object makes it difficult to meet the visibility threshold, even when using a small threshold of 25% for an item to be considered visible. Furthermore, this is the first indicator that the neural network is able to precisely identify components it is trained on even if many of those features are less than 25% visible; a full comparison of the neural networks' metrics is provided in Appendix 1 for typical CNN metrics like precision.

Outlined in Table 2, the occlusion labeling strategy resulted in observable differences in the number of Solve-

PnP observations between the two. The Xr_CC network was able to make predictions on 99.6% of images whereas the Vs_CC network achieved only 88.3%, a 29x increase. The more important factor was that the Xr_CC network was able to make predictions under the error threshold 15% more often than the network trained on only visible components; this was a substantial increase in ability. The box and whisker plots for these can be observed in Figs. 9 and 10, with detailed metrics in Table 3 and Appendix 1; outlier counts are included in Table 9.

When reporting 3D error it was determined that an observation would be an outlier if more than 10 m or 5° from truth. This approach is reasonable because Solve-PnP can behave poorly at times; when it does the system would be expected to detect and discard those observations. The system could detect poor observations for multiple reasons. First, if the orientation is multiple degrees in error, the aircraft would be expected to be flying away quickly. If the position has a large error, the aircraft may not be in the region for which the visual algorithm is supposed to make observations. Also, in a deployed system, there would be a time ordering to the observations, hence if the observation was too different than the previous few it could be discarded; it is expected the aircraft will only move approximately 10 cm max between frames during these operations.

The next interesting metric was the pixel error of the Vs_CC versus Xr_CC trained networks on finding the 3D geometric center of components. Table 4 details the findings. The network trained on only visible components had less error. This showed statistical significance via a Wilcoxon Rank-Sum test with an alpha of 0.01 and P-values significantly less than 0.001. While statistically significant, the amount was a very small fraction of a pixel difference.

While the center point pixel error is lower, the system with a model trained on only visible components performed significantly worse. The Vs_CC model was worse in both the 3D position and orientation error, shown in Figs. 9 and 10 and Table 3, as well as number of observations under the threshold, evidenced in Table 2. While this use case is slightly unique due to the predicting components of a larger rigid object, for this case it is recommended to train on the full set of components of that object regardless of occlusions in an image.

Table 2 Visibility predictions comparison

	Total predictions		Under threshold	
Xr_CC	4712	99.6%	3767	79.7%
Vs_CC	4174	88.3%	3061	64.7%

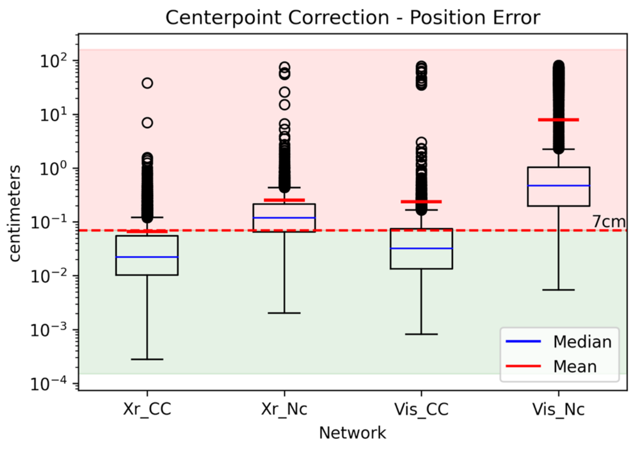

Fig. 9 Center point correction - position error

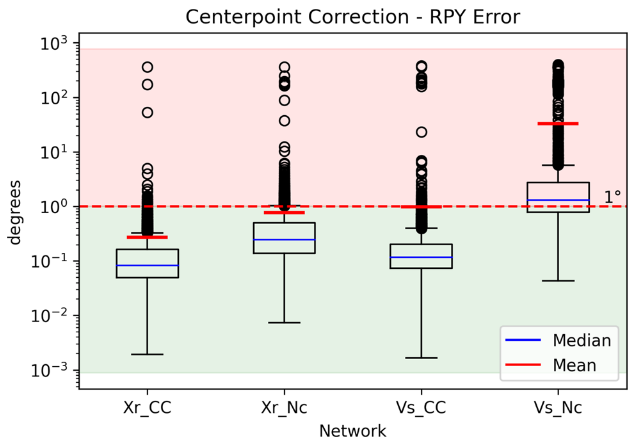

Fig. 10 Center point correction - RPY error

Table 3 Visibility pose error comparison

		All data		Under 10 m & 5°	
Xr_CC	Mean	6.61 cm	0.27°	5.64 cm	0.15°
	Median	2.22 cm	0.08°	2.22 cm	0.08°
	Std dev	57.8 cm	5.86°	10.38 cm	0.20°
Vs_CC	Mean	23.76 cm	0.98°	6.56 cm	0.19°
	Median	3.19 cm	0.12°	3.18 cm	0.12°
	Std dev	313 cm	13.53°	10.86 cm	0.25°

To summarise this analysis of training without regard for occlusions, a neural network should be trained without regard for occlusions. There is a slight degradation to average 2D pixel-precision, however the massive increase in quantity of observations allows Solve-PnP to be more performant. This was in terms of both 6D Euclidean distance as well as total images which predictions were possible for.

5.2.2 Correcting for perspective distortion labeling comparison

It is easily observable through Figs. 9 and 10 that the analysis on the truth data was confirmed. The usage of neural networks trained without correcting for perspective distortion are not able to meet the error threshold when used in the overall system. This was confirmed as statistically significant with Wilcoxon Rank-Sum tests being performed, with alpha of 0.01 and P-values significantly less than 0.001 being attained. If anything, it is impressive the Solve-PnP algorithm was able to perform so well given distorted points.

The interesting result of examining the correction made by perspective distortion is the amount the networks are capable of learning. In Fig. 11, each network's center point error is shown where that error was calculated against the labeling strategy each was trained against. This showed that the underlying neural network was able to learn both methods with similarly low error, under a pixel of error on average. Figure 12 is used to show the amount of correction the networks are applying; here the XR_CC model was evaluated against labels that were corrected and uncorrected, with the error being shown in the figure. This reveals that the models are measurably correcting for the perspective distortion, arguably indicating that perspective distortion can be learned and corrected for by a neural network.

5.2.3 Ablation study summary

In conclusion, the ablation study of neural network models had a clear result. The four models came from varying the training data to respect occlusions and if the bounding box should be corrected for perspective distortion. The Xr_CC model performed the best, this was the model trained without regard for occlusions and with perspective distortion corrected. Thus, showing that perspective distortion was measurable, correctable, and learnable by a neural network. The initial analysis quantitatively showed center point correction is required to perform within the error threshold when using object detection. Additionally, this result showed that training without regard for occlusions could improve the overall performance of the system by vastly increasing the number of observable components. This leads to other questions, such as how to best augment training data for training to correct for perspective distortion and how the network is learning to correct for the distortion.

Table 4 Center point error (pixels)

	Mean	Median	Std dev
Xr_CC	0.54	0.42	0.35
Vs_CC	0.37	0.31	0.22

5.3 Training data augmentation optimization

This section explores the results of assessing neural networks trained with different data augmentation parameters.

The number of true positives was improved with the Xr_CC_nS model versus the Xr_CC model, but was degraded for the models trained without mosaic and translation. These data sets did show statistical significance in the comparison of their true positives versus Xr_CC model. The Appendix 1 table also shows that the typical CNN metrics like F1, accuracy, precision, and recall were consistently degraded for the models trained without mosaic or translation, and were improved for the Xr_CC_nS model. The Xr_CC_nS model was significantly better than Xr_CC for all but the average number of false positives. For the use case here a small increase in false positives is acceptable. This is acceptable due to Solve-PnP's ability to maintain accuracy with false positives as well as due to observations that are made during the next and final experiment.

The model that performed the best for average 2D center point error was the one using the original hyperparameters, but without scale, Xr_CC_nS, with average values showning this improvement in Table 5. The difference was statistically significant when compared between the original model and each of the others, as well as between the Xr_CC_nS model and others; significance was shown via Wilcoxon Rank-Sum test with an alpha of 0.01, and all P-values significantly less than 0.001.

The Wilcoxon Rank-Sum tests also showed statistical significance for the improvement to the 3D position and

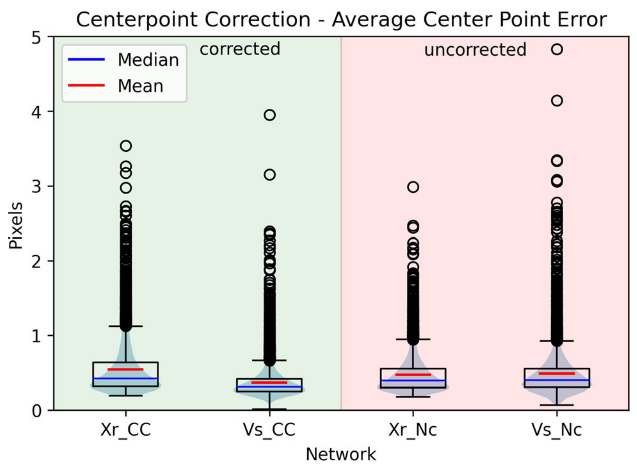

Fig. 11 Example of ability of networks to learn uncorrected and corrected bounding boxes

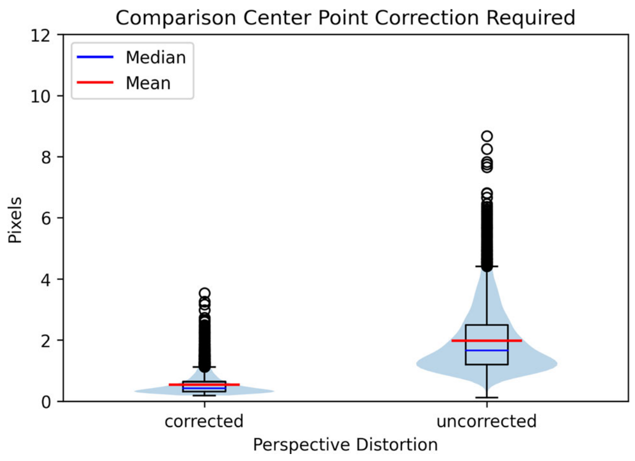

Fig. 12 Pixel amount corrected by network

orientation, with alpha of 0.01 and all P-values significantly less than 0.001. The Xr_CC_nS model showed improvement over that of the original settings for the 6D localization estimates and degradation for the other networks, evidenced in Table 5. These results indicate that the scale is the primary factor which should not be varied during training. It also showed that the mosaic and translation being removed did not show any improvement, but did degrade the detection and classification metrics, hence training should use mosaic and translation for CNNs being used for 6D pose localization.

5.4 Latent space perspective distortion experiment

With the best performing model, Xr_CC_nS, we can now examine how it is learning to correct for the error. In this experiment, there was an improvement when making predictions further away from the edge or with fewer edges for both the 2D center point error as well as a reduced number of false positives. Figures 13 and 14 show plots of the error for running the inference against the various data sets with their means detailed in Table 6; the abbreviations indicate the positions top (T), bottom (B), middle (M), left (L), and right (R) respective of the notional example in Fig. 5. The 'orig' annotation indicates the original 520 images sized at 640x480. In average 2D center point error case, the improvement showed statistical significance from a Wilcoxon Rank-Sum test with an alpha of 0.01 and all P-values significantly less than 0.001.

We needed to find that the error was not made worse by moving around the image to conclude that the network was learning the distinct appearance of an object versus the position in an image to correct for perspective distortion. In fact, in every case in Table 6, the false positives decreased

Table 5 Hyperparameter average errors comparison

	Position	Orientation	Center
	(cm)	(°)	Point (pixels)
Xr_CC	5.64	0.15	0.54
Xr_CC_nS	**5.08**	**0.13**	**0.48**
nM_nT	5.95	0.17	0.57
nM_nT_nS	6.84	0.18	0.61

and the 2D center point error decreased. This indicates the neural network is learning to correct for the perspective distortion by the distinct appearance an object has, regardless of the location in the image. Additionally, this leads to an insight about the few false positives that there are.

5.5 False positive analysis

The implication of the previous observation is that the false positives are not actually false positives. The component is there, but its center point is slightly off the edge of the image, causing it be labeled as absent in the truth data. These false positives did not have a large negative effect due to Solve-PnP with RANSAC being robust to outliers when provided enough points. This was indicated and confirmed in multiple ways.

First, the latent space experiment made the false positives disappear when there were pixels in existence to provide negative feedback, i.e., if an object was barely off the edge of the image and the neural network saw the sky background there and not the component, then it was given evidence the component was not there and made the decision to not identify it. Second, an interactive 3D plot was utilized in Python to view the position in 3D space

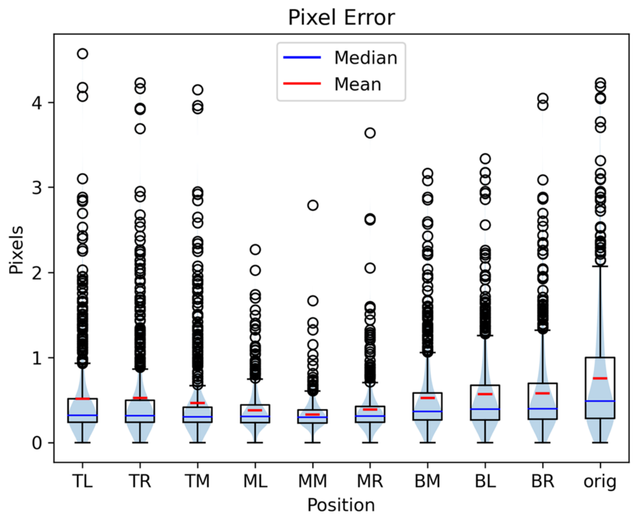

Fig. 13 Average center point error vs image location

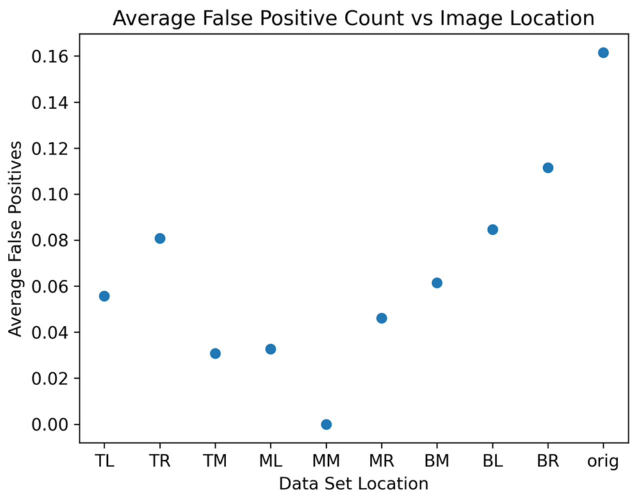

Fig. 14 Average false positive count vs image location

Table 6 Average center point errors and false positives

Position	Average center Point error(pixels)	Average false Positives
TL	0.516	0.056
TR	0.526	0.081
TM	0.465	0.031
ML	0.382	0.033
MM	0.329	0.000
MR	0.387	0.046
BM	0.527	0.062
BL	0.570	0.085
BR	0.581	0.112
Orig	0.756	0.162

where the false positives occurred. They all occurred when the aircraft was near the edge of the image.

Finally, this was confirmed with a manual verification of the points which were falsely positive. In Fig. 15 a 2D version of that 3D interactive plot was included; this is a plot of the location of the geometric center of the aircraft and where it is in the image for the entire data set, not just the latent space analysis data set. The false positives that appear towards the middle in Fig. 15 are from instances where the aircraft was near the camera, hence the geometric center of the aircraft was near the center while its components may have been near the edge. Overall, this is merely an interesting observation that may lead to future work, it does not detract from the experiments original result demonstrating the model learned the distinct appearance of the aircraft due to perspective geometry.

5.6 Running time

For 640x480 input images, the algorithm runs at 65fps on a desktop, which is more than efficient for real-time pose estimation. The desktop utilized an AMD 7950X CPU, 32GB DDR5 6400MHz Ram, a RTX 3080 GPU, and CUDA 11.7 with inference from ONNX files via OpenCV C++ API. Specifically, this implementation on average across the 5000 image test data took 3.4ms for reading an image file in, 1.4ms to size and pre-process the data, 6.8ms for forward propagation, 0.26ms for post processing the results, and 0.21ms for Solve-PnP.

6 Conclusions and future work

In summary, this study illustrated critical insights within the field of computer vision and 6D object localization. The three experiments underscored the paramount importance of correcting perspective distortion, the importance of training without regard for occlusions, the degradation of performance from image scale augmentation during training, and demonstrated translational equivariance. Thus, underscoring the need for the computer vision, robotics, and neural network communities to consider perspective geometry and distortions created by flat image sensors, essentially creating distinct appearances of objects based on their location and orientation respective to a camera.

The issue in typical object detection is the mismatch caused with the bounding box 2D center and projection of the 3D geometric center of an item being observed. This mismatch is caused by two concepts in the typical approach in object detection. The first is labeling only the visible portion of an object and the second is the perspective distortion. Each of these cause a mismatch and lead to imprecise 6D localizations. Correcting for these two items was explored in the first study.

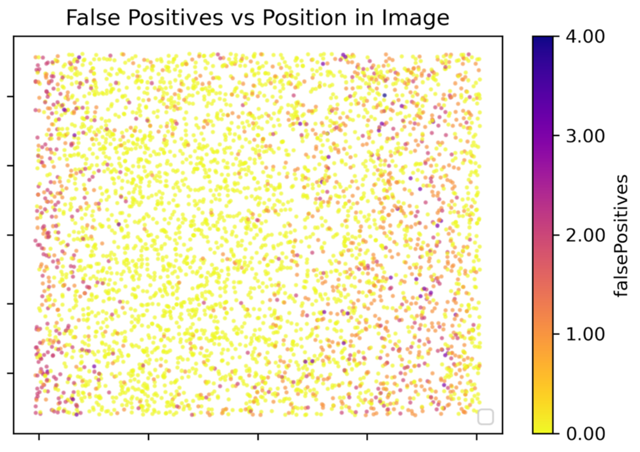

Fig. 15 False positive counts vs location on image

The first study showed that the technique of expanding a bounding box was necessary for the performance of the Solve-PnP algorithm, was learnable with near-pixel precision by the neural network, and that occluded features of the aircraft should be utilized. While their pixel-precision was slightly worse, using the occluded features ultimately resulted in a massive increase in quantity of observations and improved the 6D estimate. The second study demonstrated the training time data augmentation strategy of scale augmentation should not be used, as augmenting scale decreased the true positives found and increased the 2D error, resulting in worse 6D estimation. This study also confirmed mosaic and translation should be utilized. The final study found the model to be translationally equivariant, indicating that the distinct appearance of the item due to perspective was being learned to correct for the distortion, enforcing the Background section theory on how perspective behaves.

These results suggest that future studies could be useful to fully round out these concepts. First, the false positive analysis revealed a potential to extend the concepts featured here to the training of future neural networks by also allowing knowledge of features off the image. Future systems could also investigate other center point correction strategies like shifting the bounding boxes instead of expanding them. Those could provide additional insights to larger computer vision systems that may be able to utilize the width and height data for extremely occluded predictions or filtering of spurious results, each of which may operate better with different perspective distortion correction strategies. An eventual system could even potentially generate occlusion metrics to be used as weighting factors to create confidence, or filters for individual components' 2D precision.

For the purposes of this study, it was beneficial to use purely synthetic data. The synthetic data allowed for meticulous sub-pixel labeling, ample data sets, and extremely precise error assessments. However, these may prove to be limitations for real-world applications as such precise labels and error metrics are extremely difficult with real imagery. Additionally, if being deployed for a fleet of aircraft there is likely to be variations of textures, paint schemes, and configurations of the aircraft. This is compounded by operating aircraft not being truly rigid objects; their wings flex and control surfaces manipulate. Each of these limitations presents opportunities for future research in synthetic environments as well as with real world imagery.

While the approach shown in this article is not likely to solve issues of certifiability of a neural network for aircraft, the approach here exemplifies many of those traits: modularity, understandability, and individual testability. The system was designed with the neural network predicting human readable components of an aircraft and a separate process predicting the 6D localization. Each of the system's processes were individually assessed and the system was assessed as a whole, resulting in a system able to meet the original goal of predicting a 6D localization to less than 1° and 7 cm.

In conclusion, this study underscores the significance of considering perspective distortion in training neural networks. To optimize an object detector's performance for usage by Solve-PnP it is crucial to correct for perspective distortion, train with knowledge of occluded components, and refrain from augmenting image scale. These findings shed light on how a neural network can precisely learn and perform best in concert with Solve-PnP when armed with comprehensive knowledge, ultimately benefiting a wide array of applications in computer vision and object localization.

7 Supplementary content

This paper has an accompanying video which can be found at [5].

Appendix A: CNN Details

As a reminder of the notation used in this article, Table 7 is provided as a reference for readers summarizing the notations introduced. The abbreviation "XR" was inspired by

Table 7 List of notations

Notation	Description
Vs_CC	Only visible points with center point correction
Vs_Nc	Only visible points with NO center point correction
Xr_Nc	All points regardless of occlusions with NO center point correction
Xr_CC	All points regardless of occlusions with center point correction
Xr_CC_nS	Same hyperparameters as Xr_CC, no modification to scale
nM_nT	Same hyperparameters as Xr_CC, no mosaic or translation
nM_nT_nS	Same hyperparameters as Xr_CC, no mosaic, translation, or scale

Table 8 Training of networks

Data set	Epochs	$mAP_0.5 : 0.95$
Vs_CC	238	0.74
Vs_Nc	382	0.57
Xr_CC	1000	0.75
Xr_Nc	1000	0.72
Xr_CC_nS	1000	0.78
nM_nT	322	0.71
nM_nT_nS	385	0.71

the concept of “x-ray” vision, which was conceptually created via simulation and perfect truth data.

Metrics provided here allow comparison across all of the networks simultaneously, whereas in the body of the document only relevant information for the specific topic at hand was presented. Table 8 presents the top level validation metrics and training epochs before early stopping of 20 epochs was met. Table 9 details the various aspects of performance of these networks relevant to this application.

Appendix B: Perspective distortion

To further enable readers to develop their own intuition about how perspective distortion behaves, various figures, tables, and equations are presented here specific to a simple use case of a line segment being projected to an image plane, all in 2 dimensions. First a few scenarios are graphically examined with associated tables exploring the behavior of the center point mismatch due to perspective distortion. Then, a figure is used to define the variables relevant to these scenarios. Additional tables are then used to demonstrate the development of the equations used for calculating the error between the 2D image center and the 3D geometric center’s projection onto the image.

The first scenario examines the line segment object moving away from the camera; only the distance from the camera increases. Figure 16 shows the entire scenario, with Fig. 17 showing a zoomed in view of the image plane. Here the colors match between the figures to identify the same aspects between the figures. This shows the proportion of distortion decreases as distance increases, with values presented in Table 10.

The next scenario examines the behavior of perspective from camera rotation. Figure 18 shows how an object’s angle changes versus the image plane due to camera rotation, which vastly changing the center point mismatch, but hardly changes the size of the object on the image.

Table 9 Detailed metrics for each trained network strategy derived from test data set

Network	Average precision*	Average accuracy*	Average recall*	Average F1*	Average true positives*	Average false positives*	Average 2D average center point error (pixels)[1]			Average 3D position error (cm)[2]			Average 3D orientation error (deg)[2]			Outlier count[2]
							Mean	Mdn	σ	Mean	Mdn	σ	Mean	Mdn	σ	
Vs_CC	0.988	0.945	0.765	0.844	9.85	0.020	0.37	0.31	0.22	6.56	3.18	10.86	0.19	0.12	0.25	574
Vs_Nc	0.987	0.847	0.583	0.709	10.98	0.017	2.95	2.51	1.80	48.27	36.09	50.72	1.29	1.09	0.88	1228
Xr_Nc	0.998	0.777	0.743	0.831	28.00	0.056	2.01	1.65	1.13	17.82	11.83	21.28	0.41	0.24	0.47	32
Xr_CC	0.997	0.795	0.765	0.845	28.92	0.078	0.54	0.42	0.35	5.64	2.22	10.38	0.15	0.08	0.20	20
Xr_CC_nS	0.996	0.821	0.797	0.868	30.26	0.111	0.48	0.35	0.34	5.08	1.91	8.91	0.13	0.07	0.18	16
nM_nT	0.986	0.712	0.676	0.771	24.98	0.287	0.57	0.45	0.34	5.95	2.82	9.64	0.17	0.11	0.22	56
nM_nT_nS	0.989	0.776	0.749	0.831	28.16	0.268	0.61	0.48	0.39	6.84	3.43	11.04	0.18	0.11	0.21	16

*Metrics generated from labeling data which network was trained against; different between each

This data is evaluated against XR_CC generated truth data

This data discards outliers greater than 10 m, 5°, count in “Outlier Count” column

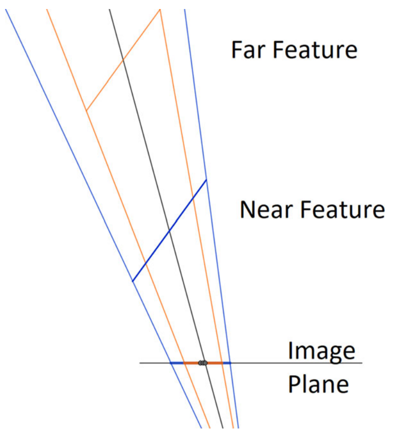

Fig. 16 Example of depth translation effect on perspective correction

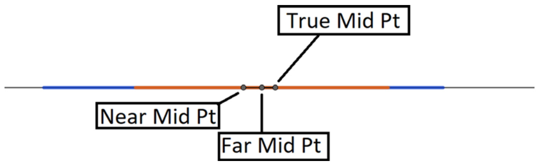

Fig. 17 Closeup of image plane and center point for close/far projections

Table 10 Distance effect on perspective distortion

Feature	Length (pixels)	Mismatch (pixels)	Mismatch (percent of length)
Near	168.95	13.47	7.97%
Far	106.91	5.68	5.31%

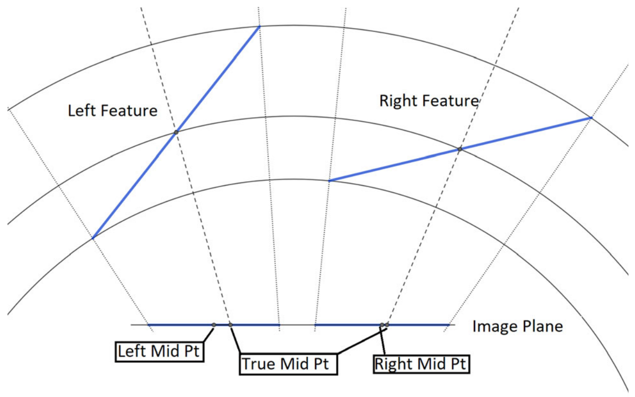

Fig. 18 Example of camera rotation effect on perspective correction

Values are provided in Table 11. This is interesting because a human eye rotating will not experience this effect due to the curve of the retina.

Table 11 Camera rotation effect on perspective distortion

Feature	Length (pixels)	Mismatch (pixels)	Mismatch (percent of length)
Left	258.29	32.83	12.71%
Right	263.62	9.87	3.74%

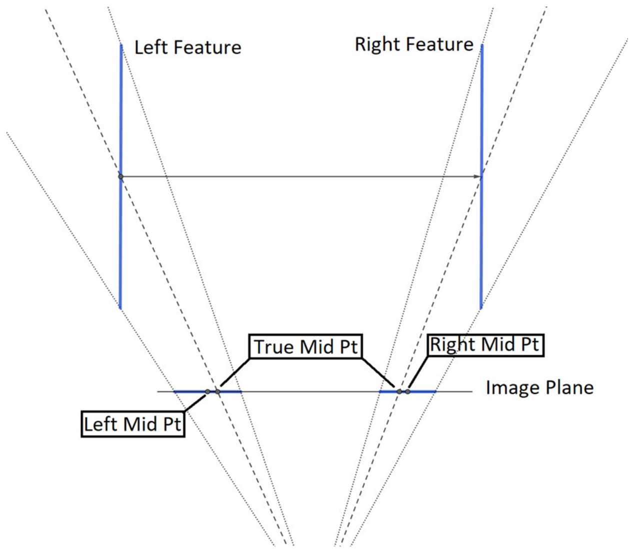

Fig. 19 Example of lateral translation effect on perspective correction

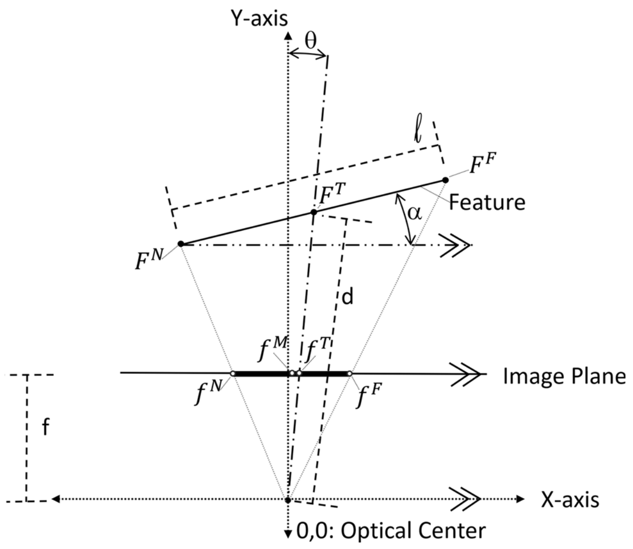

Fig. 20 Detailed diagram of potential variables for perspective distortion

The last scenario examines how an object's distortion will flip sides and changes based on where it appears in the image. This is shown in Fig. 19 with a feature moving in world space from one side of the image to the other and the mismatch switching from one side to the other.

Table 12 Image features, variables, and symbol descriptions

F^N	Left/near edge of feature
F_x	X value for F^T
F_y	Y value for F^T
F^F	Right/Far edge of Feature
F^T	True geometric center of Feature
f	Focal length in pixels, distance of optical center to image plane
f	Points on feature projection to image plane which is denoted by thick line
f^N	Projection of F^N onto image plane
f^F	Projection of F^F onto image plane
f^T	Projection of F^T onto image plane
f^M	Calculated mid point of projection on image plane: $\frac{f^N+f^F}{2}$
α	Angle Feature is rotated towards/away respective of image plane
θ	Angle F^T is from principle axis or Y-axis
d	Distance F^T is from the optical center
ℓ	Length of the feature being examined
E_{CpC}	Error or mismatch in 2D center and 3D geometric center's projection

Table 13 Perspective distortion equations describing error for center point mismatch

$F_x = d\sin(\theta)$	X value for F^T respective of θ
$F_y = d\cos(\theta)$	Y value for F^T respective of θ
$f^F = \frac{f\left(F_x+\frac{\ell}{2}\cos(\alpha)\right)}{F_y+\frac{\ell}{2}\sin(\alpha)}$	f^F respective of F_x and F_y
$f^F = \frac{fd\sin(\theta)+\frac{f\ell}{2}\cos(\alpha)}{d\cos(\theta)+\frac{\ell}{2}\sin(\alpha)}$	f^F respective of distance and θ
$f^N = \frac{f(F_x-\frac{\ell}{2}Cos(\alpha))}{F_y-\frac{\ell}{2}Sin(\alpha)}$	f^N respective of F_x and F_y
$f^N = \frac{fdsin(\theta)-\frac{f\ell}{2}Cos(\alpha)}{dcos(\theta)-\frac{\ell}{2}Sin(\alpha)}$	f^N respective of distance and θ
$f^M = \frac{f^N+f^F}{2}$	2D mid-point equation
$f^T = f tan(\theta)$	True mid-point projection equation respective of θ
$f^T = f\frac{F_X}{F_Y}$	True mid-point projection equation respective of F_x and F_y
$E_{CpC} = \left\lvert \frac{f^N+f^F}{2} - f^T \right\rvert$	Base error in center point mis-match equation
$E_{CpC} = \left\lvert \frac{f\frac{F_x}{F_y}Sin^2(\alpha)-f\cos(\alpha)Sin(\alpha)}{4\frac{F_y^2}{\ell^2}-Sin^2(\alpha)} \right\rvert$	E_{CpC} respective of F_x and F_y
$E_{CpC} = \left\lvert \frac{f tan(\theta)Sin^2(\alpha)-fCos(\alpha)Sin(\alpha)}{4\frac{d^2}{\ell^2}Cos^2(\theta)-Sin^2(\alpha)} \right\rvert$	E_{CpC} respective of distance and θ

Figure 20 presents the variables graphically for what they represent and Table 12 provides their descriptions as well as other variables with relation to those defined in the figure. Table 13 presents the fundamental equations for how the error or mismatch of the center points can be calculated. Table 14 presents a couple of the cases where the functions are minimized. The remaining figures and tables are examples of how perspective distortion and center point mismatch behaves.

Perspective distortion examples are available for reader interaction at: Fig. 16 - https://www.geogebra.org/calculator/ztsqzf5y, Fig. 18 - https://www.geogebra.org/

Table 14 Minimized locations for error in center point correction

Criteria	Resulting equation	Description
$\alpha = 0$	$\left\lvert \frac{0-0}{4\frac{F_y^2}{\ell^2}\mathrm{C}os^2(\theta)-\mathrm{S}in^2(\alpha)} \right\rvert = 0$	Minimum when $\alpha = 0$
$\alpha = 90^\circ - \theta$	$\left\lvert \frac{0-0}{4\frac{d^2}{\ell^2}\mathrm{C}os^2(\theta)-\mathrm{S}in^2(\alpha)} \right\rvert = 0$	Minimum when $\alpha = 90 - \theta$
$\lim_{d\to\infty}$	$\left\lvert \frac{\mathrm{f}tan(\theta)\mathrm{S}in^2(\alpha)-\mathrm{f}\mathrm{C}os(\alpha)\mathrm{S}in(\alpha)}{\infty} \right\rvert = 0$	Minimum as object gets further away

calculator/heseykpt, and Fig. 19 - https://www.geogebra.org/calculator/ptxpmdmc.

Acknowledgements We would like to thank our sponsors at the Naval Air Systems Command (USN/NAVAIR) and Aerospace Systems Directorate (USAF/AFRL/RQ). Your partnership and financial support made this research possible.

Data availibility The data sets generated during and/or analysed in this work are available from the corresponding author on reasonable request.

Code availability The code is not publicly available. Department of Defense organizations may contact Dr. Scott Nykl regarding inquires for use of the simulation environment and code.

Declarations

Disclaimer The views expressed are those of the author and do not reflect the official policy or position of the US Air Force, Department of Defense, or US Government.

Conflict of interest The authors have no conflicting interests to declare that are relevant to the content of this article.

References

1. Anderson James D, Nykl Scott, Wischgoll Thomas (2019) Augmenting flight imagery from aerial refueling. In: Advances in Visual Computing: 14th International Symposium on Visual Computing, ISVC 2019, Lake Tahoe, NV, USA, October 7–9, 2019, Proceedings, Part II 14, pp 154–165. Springer
2. Anderson James D, Raettig Ryan M, Larson Josh, Nykl Scott L, Taylor Clark N, Wischgoll Thomas (2022) Delaunay walk for fast nearest neighbor: accelerating correspondence matching for icp. Mach Vis Appl 33(2):31
3. Bello I, Fedus W, Du X, Cubuk ED, Srinivas A, Lin T-Y, Shlens J, Zoph B (2021) Revisiting resnets: improved training and scaling strategies. Adv Neural Inf Process Syst 34:22614–22627
4. Yannick B, Marcus V (2020) Efficientpose: an efficient, accurate and scalable end-to-end 6d multi object pose estimation approach. arXiv preprint arXiv:2011.04307,
5. Jeffrey C, Derek W, Scott N, Clark T, Brett B, Schubert KC (2023) Advancing training data techniques for 6d pose localization via object detection. YouTube video, 2023. Accessed on April 28, https://youtu.be/Ot9Ug7FAh3s
6. Dan C, Ueli M, Jürgen S (2012) Multi-column deep neural networks for image classification. In: 2012 IEEE conference on computer vision and pattern recognition, pp 3642–3649. IEEE
7. Alberto C, Rad M, Verdie Y, Moo YK, Pascal F, Vincent L (2017) Robust 3d object tracking from monocular images using stable parts. IEEE Trans Pattern Anal Mach Intell 40(6):1465–1479
8. Ekin CD, Barret Z, Dandelion M, Vijay V, Quoc V Le (2019) Autoaugment: Learning augmentation strategies from data. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 113–123
9. Ekin CD, Barret Z, Jonathon S ,Le Quoc V (2020) Randaugment: practical automated data augmentation with a reduced search space. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops, pp 702–703
10. Paolo Di F, Dal MC, Kinh T, Stefano M (2018) Kcnn: extremely-efficient hardware keypoint detection with a compact convolutional neural network. In: Proceedings of the IEEE conference on computer vision and pattern recognition workshops, pp 682–690
11. Ding X, Li Q, Cheng Y, Wang J, Bian W, Jie B (2020) Local keypoint-based faster r-cnn. Appl Intel 50:3007–3022
12. Golnaz G, Yin C, Aravind S, Rui Q, Lin T-Y, Ekin CD, Le Quoc V, Barret Z (2021) Simple copy-paste is a strong data augmentation method for instance segmentation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 2918–2928
13. Girshick R (2015) Fast r-cnn. In: Proceedings of the IEEE international conference on computer vision, pp 1440–1448
14. Joseph H, Glyn R, Nassib N, Roger A, Myers L, McCormick J (2006) Darpa autonomous airborne refueling demonstration program with initial results. In: Proceedings of the 19th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS 2006), pp 674–685
15. He K, Gkioxari G, Dollár P, Girshick R (2017) Mask r-cnn. In: Proceedings of the IEEE international conference on computer vision, pp 2961–2969

16. He K, Zhang X, Ren S, Sun Jian (2015) Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE Trans Pattern Anal Mach Intel 37(9):1904–1916
17. Yisheng H, Wei S, Haibin H, Jianran L, Haoqiang F, Jian S (2020) Pvn3d: A deep point-wise 3d keypoints voting network for 6dof pose estimation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 11632–11641
18. Donald CDR, Costello III H, Adams Richard (2021) Framework for certification of autonomous systems within naval aviation a white paper
19. Jocher Glenn , Stoken Alex, Borovec Jirka,, ChristopherSTAN, Liu Changyu NanoCode012, Laughing, tkianai, Adam Hogan, lorenzomammana, yxNONG, AlexWang1900, Laurentiu Diaconu, Marc, wanghaoyang0106, ml5ah, Doug, Francisco Ingham, Frederik, Guilhen, Hatovix, Jake Poznanski, Jiacong Fang, Lijun Yu, changyu98, Mingyu Wang, Naman Gupta, Osama Akhtar, PetrDvoracek, and Prashant Rai. ultralytics/yolov5: v3.1 - Bug Fixes and Performance Improvements, October 2020
20. Kehl W, Manhardt F, Tombari F, Ilic S, Navab N (2017) Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again. In: Proceedings of the IEEE international conference on computer vision, pp 1521–1529, SSD 6D
21. Kurdthongmee W, Kurdthongmee P, Suwannarat K, Kiplagat JK (2022) A yolo detector providing fast and accurate pupil center estimation using regions surrounding a pupil. Emerg Sci J 6(5):985–997
22. Le Tuan-Tang, Le Trung-Son Yu-Ru, Chen Joel Vidal, Lin Chyi-Yeu (2021) 6d pose estimation with combined deep learning and 3d vision techniques for a fast and accurate object grasping. Robot Auton Syst 141:103775
23. Liu L, Campbell D, Li H, Zhou D, Song X, Yang R (2020) Learning 2d-3d correspondences to solve the blind perspective-n-point problem. arXiv preprint arXiv:2003.06752
24. Liu W, Qian B, Yu S, Tao M (2022) Recent advances of monocular 2D and 3D human pose estimation: a deep learning perspective. ACM Comput Surv 55(4):1–41
25. Lowe David G (2004) Distinctive image features from scale-invariant keypoints. Int J Comput Vis 60:91–110
26. James CL (2022) Monocular pose estimation for automated aerial refueling via perspective-n-point. Technical report, Air force institute of technology Wright–Patterson AFB OH WRIGHT-PATTERSON ,
27. Team Mighty (2022) The first-ever mid-air refueling happened in 1923 between biplanes, Dec
28. Minderer M, Gritsenko A, Stone A, Neumann M, Weissenborn, Alexey D, Dosovitskiy, Mahendran A, Arnab A, Dehghani M, Shen Z et al. (2022) Simple open-vocabulary object detection. In: European Conference on Computer Vision, pp 728–755. Springer
29. Nangia RK (2007) 'Greener' civil aviation using air-to-air refuelling - relating aircraft design efficiency and tanker offload efficiency. Aeronaut J 111(1123):589–592
30. Nykl S, Mourning C, Leitch M, Chelberg D, Franklin T, Liu C (2008) An overview of the steamie educational game engine. In: 2008 38th Annual Frontiers in Education Conference, pp F3B–21. IEEE
31. Park K, Patten T, Vincze M (2019) Pix2pose: pixel-wise coordinate regression of objects for 6d pose estimation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp 7668–7677
32. Park TH, D'Amico S (2023) Robust multi-task learning and online refinement for spacecraft pose estimation across domain gap. Adv Space Res. https://doi.org/10.1016/j.asr.2023.03.036
33. Parry Jonathon, Hubbard Sarah (2023) Review of sensor technology to support automated air-to-air refueling of a probe configured uncrewed aircraft. Sensors 23(2):995
34. Peng S, Liu Y, Huang Q, Zhou X, Bao H (2019) Pvnet: pixel-wise voting network for 6dof pose estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 4561–4570
35. Periyasamy AS, Amini A, Tsaturyan V, Behnke S (2023) Yolopose v2: understanding and improving transformer-based 6d pose estimation. Robot Auton Syst 168:104490
36. Rad M, Lepetit V (2017) Bb8: a scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth. In: Proceedings of the IEEE international conference on computer vision, pp 3828–3836
37. Redmon J, Divvala S, Girshick R, Farhadi A (2016) You only look once: unified, real-time object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 779–788
38. Redmon J, Farhadi A (2017) Yolo9000: better, faster, stronger. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 7263–7271
39. Redmon J, Farhadi A (2018) Yolov3: an incremental improvement. arXiv preprint arXiv:1804.02767
40. Ren S, He K, Girshick R, Sun J (2015) Faster r-cnn: towards real-time object detection with region proposal networks. Advances in neural information processing systems, 28
41. Rukhovich D, Vorontsova A, Konushin A (2022) Fcaf3d: fully convolutional anchor-free 3d object detection. In: European Conference on Computer Vision, pp 477–493. Springer
42. Sattler Torsten, Leibe Bastian, Kobbelt Leif (2016) Efficient & effective prioritized matching for large-scale image-based localization. IEEE Trans Pattern Anal Mach Intell 39(9):1744–1756
43. Schönberger JL, Pollefeys M, Geiger A, Sattler T (2018) Semantic visual localization. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 6896–6906
44. Schweikhard K (2008) Results of nasa/darpa automatic probe and drogue refueling flight test. Technical report
45. Sermanet P, Eigen D, Zhang X, Mathieu M, Fergus R, LeCun Y (2013) Overfeat: integrated recognition, localization and detection using convolutional networks. arXiv preprint arXiv:1312.6229
46. Steiner Andreas, Kolesnikov Alexander, Zhai Xiaohua, Wightman Ross, Uszkoreit Jakob, Beyer Lucas (2021) How to train your vit? data, augmentation, and regularization in vision transformers. arXiv preprint arXiv:2106.10270
47. Tekin B, Sinha SN, Fua P (2018) Real-time seamless single shot 6d object pose prediction. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 292–301
48. Tyszkiewicz MJ, Maninis K-K, Popov S, Ferrari V (2022) Raytran: 3d pose estimation and shape reconstruction of multiple objects from videos with ray-traced transformers. In: European Conference on Computer Vision, pp 211–228. Springer
49. Vidal J, Lin C-Y, Lladó X, Martí R (2018) A method for 6d pose estimation of free-form rigid objects using point pair features on range data. Sensors 18(8):2678
50. Wang C, Xu D, Zhu Y, Martín-Martín R, Lu C, Fei-Fei L, Savarese S (2019) Densefusion: 6d object pose estimation by iterative dense fusion. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 3343–3352
51. Wu Y, Zand M, Etemad A, Greenspan M (2022) Vote from the center: 6 dof pose estimation in rgb-d images by radial keypoint voting. In: European Conference on Computer Vision, pp 335–352. Springer
52. Xiang Y, Schmidt T, Narayanan V, Fox D (2017) Posecnn: a convolutional neural network for 6d object pose estimation in cluttered scenes. arXiv preprint arXiv:1711.00199

53. Zand M, Etemad A, Greenspan M (2022) Objectbox: from centers to boxes for anchor-free object detection. In: European Conference on Computer Vision, pp 390–406. Springer
54. Zhang F, Gao J, Song C, Zhou H, Zou K, Xie J, Yuan T, Zhang J (2023) Tpmv2: an end-to-end tomato pose method based on 3D key points detection. Comput Electron Agric 210:107878
55. Zhang Xin, Jiang Zhiguo, Zhang Haopeng (2019) Real-time 6d pose estimation from a single rgb image. Image Vis Comput 89:1–11
56. Zhang Xin, Jiang Zhiguo, Zhang Haopeng (2020) Out-of-region keypoint localization for 6d pose estimation. Image Vis Comput 93:103854
57. Zhang Yu, Guo Zhongyin, Jianqing Wu, Tian Yuan, Tang Haotian, Guo Xinming (2022) Real-time vehicle detection based on improved yolo v5. Sustainability 14(19):12274
58. Zhu X, Su W, Lu L, Li B, Wang X, Dai J (2020) Deformable detr: Deformable transformers for end-to-end object detection. arXiv preprint arXiv:2010.04159
59. Zoph B, Cubuk ED, Ghiasi G, Lin T-Y, Shlens J, Le Quoc V (2020) Learning data augmentation strategies for object detection. In Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVII 16, pp 566–583. Springer

Authors and Affiliations

Jeffrey Choate[1] · Derek Worth[1] · Scott Nykl[1] · Clark Taylor[1] · Brett Borghetti[1] · Christine Schubert Kabban[2]

✉ Jeffrey Choate
jeffrey.choate@us.af.mil

Derek Worth
Derek.Worth.2@spaceforce.mil

Scott Nykl
Scott.Nykl@afit.edu

Clark Taylor
Clark.Taylor@afit.edu

Brett Borghetti
Brett.Borghetti@afit.edu

Christine Schubert Kabban
Christine.Schubert@afit.edu

[1] Department of Electrical and Computer Engineering, Air Force Institute of Technology, Wright-Patterson AFB, Dayton, OH 45433, USA

[2] Department of Mathematics and Statistics, Air Force Institute of Technology, Wright-Patterson AFB, Dayton, OH 45433, USA