

# From Simulation to Reality: Transfer Learning for Automating Pseudo-Labeling of Real and Infrared Imagery

Jeffrey Choate,\* Derek Worth, Scott L. Nykl, Clark Taylor, Brett Borghetti, Christine Schubert Kabban, and Ryan Raettig

Training a convolutional neural network (CNN) for real-world applications is challenging due to the requirement of high-quality labeled imagery. This study employs pseudo-labeling and transfer learning, built upon a 6D pose estimation framework. A CNN trained on synthetic images predicts bounding boxes (bbox) for an object's components in a real image. With as few as four bbox predictions, the framework solves for the object's pose relative to the camera and reprojects bboxes for all components onto that image. The pose and reprojections allow filtering of bad predictions, a common issue in pseudo-labeling. Thereby, enabling automated labeling of large datasets with minimal human intervention. Tested on color and long-wave infrared imagery captured during December 2023 flight tests, this process demonstrates increased predictions, enhanced performance across situations, reduced reprojection error, and stabilized pose predictions. This technique is significant as it enables labeling of real-world imagery without expensive truth systems, requiring only a camera. It supports learning and labeling of previously captured imagery without known camera calibrations, facilitating labeled data creation for impractical-to-simulate sensors. Ultimately, this transfer learning approach provides a low-cost and precise method for creating CNNs trained on operationally relevant data, previously unattainable by the everyday user.

high-quality, precisely labeled imagery, especially in the domain of long-wave infrared (LWIR) imagery. This scarcity not only affects fields like robotics, computer vision, and automation but also has tangible implications for critical operations for the United States Air Force (USAF), such as aerial refueling. Aerial refueling facilitates global reach for military operations and expands many aircrafts' mission capabilities. The development of automated aerial refueling (AAR) represents extending these capabilities to unmanned aircraft as well as increasing the safety of manned aircraft. Traditional techniques for developing imagery used to train a convolutional neural network (CNN) for this automation rely on error-prone human data annotation, expensive sensors, or simulations, which often fall short in capturing the real-world complexities and nuances, particularly in different wavelengths. Thus, a large challenge is posed when attempting to automate and transition such algorithms out of the lab setting to real-world operations.

## 1. Introduction


The development of robust automation algorithms reliant on vision-based sensors faces a significant hurdle: the scarcity of

This study solves the lack of operationally realistic data with the development of a novel transfer learning system to automate the creation of high-quality labels for such data. The approach leverages pseudo-labeling and transfer learning with a 6 degree of freedom (6DoF) pose estimation system. This enables the labeling of real images, including LWIR imagery, without the need for costly truth sources or labor intensive, error-prone manual labeling of large datasets. On operational imagery directly from real-world flight tests, this study shows that pixel reprojection error is reduced and bounding box (bbox) estimate counts are increased, resulting in a higher number of well-estimated 6DoF poses and fewer conspicuous errors in predictions, known as bad pose estimates.

To summarize the pseudo-labeling technique: a 3D model of an object is used in the OpenGL simulation environment, AfrBurner,<sup>[1,2]</sup> to generate perfectly labeled images of a known 3D object. These images are used to train a YOLOv5<sup>[3]</sup> CNN to draw bboxes around various parts of the object; typically 20 or more parts of the object will have associated bboxes for them. Then the Solve Perspective-N-Point (Solve-PnP)<sup>[4]</sup> algorithm estimates the 6DoF pose (position and orientation) of the object, if it

J. Choate, D. Worth, S. L. Nykl, C. Taylor, B. Borghetti, R. Raettig  
Department of Electrical and Computer Engineering  
Air Force Institute of Technology  
Wright-Patterson AFB, OH 45433, USA  
E-mail: jeffrey.choate@us.af.mil

C. Schubert Kabban  
Department of Mathematics and Statistics  
Air Force Institute of Technology  
Wright-Patterson AFB, OH 45433, USA

 The ORCID identification number(s) for the author(s) of this article can be found under <https://doi.org/10.1002/aisy.202400575>.

© 2025 The Author(s). Advanced Intelligent Systems published by Wiley-VCH GmbH. This is an open access article under the terms of the Creative Commons Attribution License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited.

DOI: 10.1002/aisy.202400575

receives at least four bbox estimates from YOLO. This 6DoF pose can then be used to reproject all the bboxes desired for a given object back into an image; hence, with as little as four YOLO predictions, reprojection can label all 20+ bboxes for an image.

Training on pseudo-labeled data is often hindered by the necessity for high-quality data. This requires identifying and mitigating false positives, false negatives, and noisy or inaccurate data. Our system solves this issue by utilizing the pose, reprojection error, and number of predictions to filter bad pose estimates. Thus, with minimal human supervision, this process can be automated for hundreds of thousands of real images. It is worth noting that typically the authors prefer to use five or more features for Solve-PnP to enhance its stability by leveraging the random sample consensus algorithm (RANSAC).

To demonstrate the effectiveness of this technique, multiple real datasets from a December 2023 flight test event are used. The objects labeled and predicted for this study were a Calspan Learjet C25D and a United States Navy (USN) refueling drogue basket. The flight test consisted of two rear-facing LWIR cameras mounted on the tanker aircraft, a Gulfstream G-III, and a color electro-optical (EO) camera mounted in the cockpit of the receiver aircraft, a Calspan Learjet C25D. This study restricts reporting metrics to two of the flights. This resulted in excess of 200,000 images for both LWIR and color imagery that could be assessed. Flight 1 is used as the training and validation dataset, holding back Flight 2 as a test dataset; there were two LWIR cameras on the tanker aircraft, the “left” of Flight 1 was the training and validation dataset.

Thus, three different CNNs were trained, summarized in **Table 1**. The EO\_Drogue network was trained on color simulation imagery with the drogue facing the camera. Then used to pseudo-label the real EO color imagery from the Learjet cockpit. The LWIR\_Learjet and LWIR\_Drogue networks were trained on simulation imagery converted to grayscale, to better imitate the appearance of LWIR. The drogue was facing away from the camera and the Learjet was facing the camera in this training imagery. These networks were used to pseudo-label the LWIR imagery captured from the G3.

Additionally, an experiment was conducted to assess the impact of camera intrinsics on the pseudo-labeling process. While precise camera intrinsics are critical for accurate 6DoF pose estimates, their significance in the pseudo-labeling process proposed in this article is negligible. This implies that approximate camera intrinsics and field of view (FoV) estimates can suffice for applying the pseudo-labeling technique. This demonstrates that our pseudo-labeling technique can be applied to older datasets with limited information about camera calibration, thereby simplifying label creation for object detection teams.

The remainder of this article is organized as follows. Section 2 provides an overview of related work in the field of AAR,

pseudo-learning, and transfer learning techniques. Section 3 outlines the methodology employed in our study, including details of the flight tests and data collection procedures. Section 4 presents the experiment results, followed by a discussion of their implications. Finally, Section 5 offers concluding remarks and outlines directions for future research. A video showing these findings is available at.<sup>[5]</sup>

The contributions of the paper are: 1) A novel transfer learning technique that facilitates semi-autonomous labeling of real images, including LWIR imagery, without the need for expensive truth sources. 2) Experimental results demonstrating increased performance and prediction capabilities of CNN and 6DoF predictions trained on the semi-autonomously labeled real-world images. 3) Demonstration of proposed technique without regard for camera calibration, revealing applicability of technique to datasets without quality knowledge about the camera used.

## 2. Background and Related Work

### 2.1. Aerial Refueling

Aerial refueling is critical to military operations as it extends the mission capabilities and range of aircraft, allowing the United States to have global reach. For aerial refueling, there are two main scenarios: USAF and USN or North Atlantic Treaty Organization (NATO). In both scenarios there is a tanker aircraft carrying large amounts of fuel and a receiver aircraft, which desires that fuel. In the USAF scenario, the receiver has a refueling receptacle, a hole, that the tanker must maneuver a flying boom with a telescoping tube into. In this scenario the receiver must find and maintain position behind the tanker while the tanker must identify the refueling receptacle and manipulate its boom into it. The USN or NATO scenario differs in that the tanker simply extends a drogue basket behind it and the receiver must find and connect itself to that basket. In both scenarios, aircrew must take months to train until they can use their vision to estimate the position of the opposing aircraft, and in the USN case, identify a small drogue basket as well.<sup>[6]</sup>

There have been limited successes at past attempts to automate the aerial refueling process.<sup>[7,8]</sup> They have involved expensive custom-built sensor suites, causing these techniques to be restricted to demonstrations. Past research tended toward stereo-vision as an approach;<sup>[9,10]</sup> while it showed limited success, it proved untenable to overcome the noisy nature of the stereo-vision system compared to more recent approaches. Recent research<sup>[11–14]</sup> revealed that off the shelf cameras paired with systems using computer vision and CNNs are able to replace these previously used, expensive, and custom-built systems. However, this recent research has not yet shown a reliable transfer to real-world imagery from simulated imagery.

**Table 1.** Summary of dataset parameters.

Dataset	Range [m]	Horizontal	Vertical	Roll	Pitch	Yaw
LWIR Learjet	(28, 55)	(−80%, 80%)	(−75%, −30%)	(−2°, 2°)	(−36°, −28°)	(−2°, 2°)
LWIR Drogue	(2, 16)	(−50%, 50%)	(−70%, 25%)	(0°, 0°)	(−25°, −15°)	(172°, 188°)
EO Drogue	(5, 30)	(−60%, 60%)	(−10%, 50%)	(0°, 0°)	(−15°, 0°)	(−5°, 5°)

## 2.2. Labeling Methods

Labeling techniques for object detection can be categorized into four main types: manual labeling, synthetic labeling, truth system-assisted labeling, and pseudo-labeling. These techniques are often used in combination to create highly effective object detectors.

### 2.2.1. Manual Labeling

Manual labeling involves human users defining bboxes around objects in images, which are then saved as training data for CNNs. In this process, specialized software is used to draw bboxes around objects in images, creating essential labeled data for CNN training. Several software suites and services facilitate this task.

One prominent tool is Amazon Mechanical Turk,<sup>[15]</sup> a crowd-sourcing marketplace where datasets can be human-labeled. Other notable options include RoboFlow,<sup>[16]</sup> Amazon SageMaker,<sup>[17]</sup> LabelMe,<sup>[18]</sup> and Labellerr.<sup>[19]</sup> These software suites enable users to draw bboxes, interpolate across frames in a video,<sup>[20]</sup> and potentially integrate with other CNNs to assist with labeling the data.

ImageNet,<sup>[21]</sup> one of the most prominent object detection datasets, was created with the LabelMe software manually. While manual labeling has significantly benefited the computer vision community, it is not without its challenges, including issues with precision in bboxes, false positives, false negatives, as well as being time-consuming and expensive.<sup>[22,23]</sup>

### 2.2.2. Synthetic Labeling

To overcome the limitations of manual labeling and capitalize on recent advancements in computer graphics, synthetic labeling has gained prominence. This approach utilizes 3D graphics engines, also known as simulators, to rapidly and inexpensively generate precisely labeled datasets. Several graphics engines and software suites facilitate this process.

Notable graphics engines include Blender,<sup>[24]</sup> Unreal Engine,<sup>[25]</sup> Unity,<sup>[26,27]</sup> and AftBurner.<sup>[2,9,12]</sup> These platforms enable users to generate data quickly and cost-effectively. They offer capabilities such as variable lighting conditions, material effects, customizable camera parameters, and precise 3D and 2D object knowledge within images. While this study utilizes AftBurner, the other tools are also viable options for implementing the techniques proposed in this article.

Research by Hattori et al.<sup>[28]</sup> demonstrated that a CNN trained on synthetic imagery can outperform a CNN trained on a very limited real dataset. Various studies<sup>[29–31]</sup> have demonstrated that CNNs trained with synthetic image generation techniques can achieve significant success in transfer learning to real imagery.

Similar to manual image labeling, these tools and techniques have immensely benefited the computer vision community. However, they cannot perfectly replicate real-world conditions or accurately simulate all types of sensors, such as LWIR cameras, creating a domain gap that poses challenges for CNNs to overcome.

### 2.2.3. Truth System Labeling

A straightforward approach to mitigate the domain gap between synthetic and real imagery is to use real imagery itself. However, achieving precise labels for object detection in real imagery is critical. One effective method involves leveraging additional truth systems and sensors capable of automating label creation in real-world scenarios, thereby ensuring the required precision.

Several systems facilitate this process, including motion capture (MoCap), differential global positioning system (GPS), light detection and ranging (Lidar), stereo vision, and fiducial markers. MoCap systems like Vicon,<sup>[32]</sup> Phasespace,<sup>[33]</sup> and OptiTrack<sup>[34]</sup> excel in precisely locating and tracking items and their features in 3D space. However, they can be costly and impose limitations on the size and distance of objects they can accurately track.

Differential GPS enables precise positioning between objects within a scene and has been successfully employed in tasks like aerial refueling.<sup>[35]</sup> Nonetheless, its implementation requires multiple installations across the scene, can be expensive, demands careful calibration, and objects in the scene can occlude the satellite signals.<sup>[36,37]</sup> Other approaches utilize Lidar<sup>[38]</sup> or stereo vision<sup>[39,40]</sup> to get positions of objects in a scene.

Additionally, Worth et al.<sup>[41]</sup> demonstrated success using fiducial markers to localize a camera and objects within a MoCap room, achieving precise labeling in that controlled environment. Similarly, Kiyokawa<sup>[42]</sup> employed fiducial markers in images to automate low-precision labels for nearby objects, then masking the markers to prevent the CNN from learning based on them.

Despite their effectiveness, these truth systems share common limitations: they are constrained by the environments in which they can be applied and often involve significant implementation costs.

### 2.2.4. Pseudo-Labeling

The technique of pseudo-labeling has emerged to address the challenges posed by the other labeling methods while still utilizing real imagery. This approach generally involves using a CNN to generate inferred labels for a dataset, which are then used to further train the same or a completely new CNN.<sup>[43]</sup> This iterative process continues until the desired performance is achieved. A key concept in pseudo-labeling research is the necessity for high-quality labels. However, the outputs of CNNs typically are noisy, contain false positives, and false negatives.<sup>[44–49]</sup> Consequently, a significant area of research focuses on developing custom loss functions to reduce the penalties for false negatives, manage the penalty of noisy data, and allow for soft labels.<sup>[39,50–52]</sup> Soft labels are labels where a bbox is assigned a probability distribution over multiple classes, allowing it to belong to more than one class with varying degrees of confidence.

Another significant area of research in the field of pseudo-labeling is data filtering. Various research efforts have explored different techniques to achieve this, including clustering the outputs to filter out poor estimates,<sup>[53,54]</sup> utilizing additional sensors for filtering,<sup>[38]</sup> and applying consistency regularization, which involves perturbing the data and assessing the consistency of the outputs.<sup>[39,55,56]</sup> Other methods involve using confidence

scores reported by the CNN<sup>[47,49]</sup> or employing alternative filtering techniques.<sup>[46,57,58]</sup>

Curriculum learning is another technique used by the pseudo-labeling field, where initial training focuses on simpler examples, progressively advancing to more complex and challenging images.<sup>[45,50,53,59]</sup> Pseudo-labeling techniques have demonstrated success across various imaging spectrums in object detection applications.<sup>[55,57,60,61]</sup>

Pseudo-labeling serves as the primary technique investigated in this study. However, our contribution is due to the novel employment of the Solve-PnP algorithm and utilizing knowledge of the 3D world for filtering bad pose estimates. These are critical innovations within the pseudo-labeling field. Ultimately, these labeling techniques aim to enhance transfer learning, improving the efficacy of CNNs in real-world imagery applications.

### 2.3. Transfer Learning

Transfer learning, as defined by Torrey and Shavlik,<sup>[62]</sup> “the improvement of learning in a task through the transfer of knowledge from a related task that has already been learned.” Pan<sup>[63]</sup> summarizes it as “aiming to boost performance of the target domain by utilizing the source domain.” Applications of transfer learning range from training a CNN as a classifier and then attaching more layers to further train it as an object detector, as seen in YOLOv5,<sup>[3]</sup> to training a CNN on cars and then utilizing it to find trucks or further train on datasets of trucks.<sup>[63–65]</sup>

For the study herein, transfer learning involved using a CNN trained on synthetic color-rendered images of 3D scanned objects and applying those CNNs to real EO images or real LWIR images. Transfer learning is not restricted to CNNs and extends to other machine learning techniques, but the examples provided herein are limited to CNNs.

Research indicates that optimal CNN training involved a blend of real and synthetic data,<sup>[29,66]</sup> balancing the richer contextual information in real data with the cleaner, precisely labeled synthetic data.<sup>[29,67]</sup> Thus, the ability of techniques like pseudo-labeling to generate precise labels holds promise for bridging the gap in transfer learning, enabling CNNs to perform effectively on real-world imagery and scenarios.

### 2.4. Pose Estimators

A crucial component of the methodology presented in this article is the ability to accurately estimate the 6DoF pose of objects. This capability has been extensively explored in prior research, particularly highlighted in studies on AAR at the Air Force Institute of Technology.<sup>[11–14]</sup> Key details regarding these techniques are provided in Section 3.

Beyond the specific approach outlined here, several other methods for object pose estimation exist, which could potentially complement and benefit from the concepts discussed in this article, particularly in the context of data labeling. For instance, the BB8 method<sup>[68]</sup> employs a CNN to predict the front and rear bboxes for an object, using Solve-PnP to compute the pose based on the corners of the bboxes. Another approach, as seen in,<sup>[69]</sup> involves one CNN selecting a sub-region from an image and another CNN predicts the pose on that sub-region.

PoseCNN,<sup>[70]</sup> on the contrary, directly predicts the pose using a CNN. Various other pose estimation networks with nuanced variations also exist.<sup>[71–75]</sup>

It’s important to note that the examples mentioned earlier focus on approaches that do not utilize depth sensors, aligning with the scope of this article. However, these methods offer potential compatibility with the concepts presented herein, particularly in their application toward enhancing the efficiency and accuracy of image data labeling processes.

## 3. Methodology

This section discusses the nuances of the methodology required to execute the pseudo-labeling process. The overall process can be summarized as the following set of steps: 1) Create 3D model of object, 2) Generate synthetic imagery and labels, 3) Train CNN on synthetic imagery, 4) Pseudo-label real images, 5) Filter pseudo-labeled images, 6) Train on pseudo-labeled images, 7) Measure improvement, and 8) Repeat Steps 4–7.

### 3.1. 3D Object Creation

In order to generate imagery, the initial step involves creating 3D object files suitable for rendering. For this project, two objects were developed: a drogue basket with coupler and a Learjet 25D.

To create the drogue object, an Artec Leo<sup>[76]</sup> was used to 3D scan it. The drogue was removed from the hose and placed on the ground with the basket facing up for scanning. This allowed the basket to flair out similarly to how it would during flight. The 3d scan was then processed with Artec’s software. This processing involved aligning scans, removing artifacts, and filling holes in the scan, resulting in a realistic representation of the drogue, as depicted in **Figure 1**.

The Learjet could not be fully scanned due to its size, so only the front nose cone was scanned and processed with the Artec Leo scanner. Although additional scanners capable of scanning large objects exist, they were not available for this experiment. Blender<sup>[24]</sup> was then used to merge this scan with an .obj model of the aircraft provided by Coherent Technical Services Inc.



**Figure 1.** Artec Studio 17 Render of Drogue Model.



(CTSI), as shown in **Figure 2**. This model, created from computer-aided design software, was sized according to engineering design documents for the Learjet. The Learjet model lacked associated textures, but this was not a significant concern for the experiments in this article because the Learjet model was used with LWIR imagery, which does not display textures as observed with EO color cameras. In both cases, the result was an .obj file loaded into the AftBurner graphics engine for rendering in the next step.

### 3.2. Synthetic Imagery and Label Creation

These 3D .obj files were then used to generate imagery via the 3D graphics engine AftBurner. The techniques employed here are similar to past work,<sup>[11]</sup> summarized as follows: 3D point clouds defining components of the object are defined in their local coordinate frame, projective geometry is used to get 2D pixel coordinates from each of those points, bboxes are then fit around the projected points for each component, thus resulting in a labeled image. Because bboxes are used instead of key points, it is necessary to correct for perspective in the 2D screen projection. This correction is achieved by projecting the 3D geometric center of each component and then expanding each bbox such that the center of the 2D bbox matches the projection of the 3D center. Additionally, all components are projected and labeled regardless of their visibility.

This resulted in three files of point clouds being used to define components on the drogue and Learjet. There were two unique files for the drogue since it is being viewed from two vastly different angles and the same features likely wouldn't be as apparent from different perspectives; this is why Table 1 has three entries. This allowed the generated imagery and labels to be as realistic and specific to the intended use cases. The Learjet had 24 component point clouds defined for it and the drogue had 13 component point clouds defined for it, in each of its files. These point clouds tended toward human readable component names, that is, "U" on the drogue, or "left wing" on the Learjet, however the features were not always discrete components, that is, "topFins" on the drogue or "TailMid" on the

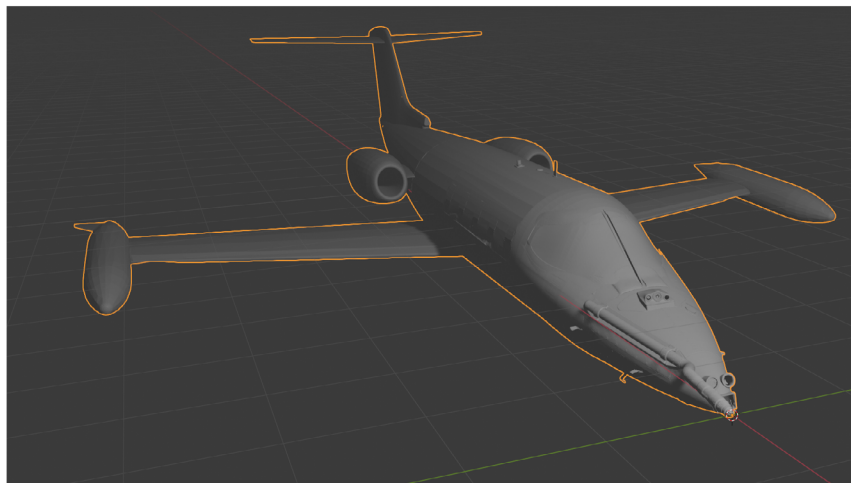
Learjet. This was seen as acceptable because the receptive field of the YOLO CNN is able to see more of the image than what is precisely within a bbox in order to determine where the bbox should be positioned.

The background skybox is randomly chosen amongst 25 images, lighting position is randomized, and the objects are randomly placed in the camera's frustum within a set of pose parameters described in Table 1. The main parameters here are the range from the camera, the horizontal and vertical limits the center of the object may appear in the image, and the roll, pitch, and yaw limits of the object itself; these are respective of the 3D .obj files created earlier with the intent of displaying the amount of variation used.

Due to the drogue's symmetry, its pose was treated with special care to ensure the components being labeled, that is, bboxes, were all rotationally invariant, while still rotating the object being displayed. This was accomplished by randomly setting the 6DoF pose for the object for all but the rotation about the X-axis; the rotation about the X-axis was always set the same as the tanker aircraft the camera was mounted on. The components were then projected to the screen for bbox creation and labeling, then the drogue was rotated randomly up to 360° about the X-axis before it was rendered. This ensured the bboxes were always rotationally invariant in the generated images and the drogue appeared more realistically.

### 3.3. CNN Training on Synthetic Data

There were three dataset scenarios used for these experiments: the LWIR Learjet, LWIR drogue, and EO drogue. For each dataset, 10 000 training images and 3,000 validation images were synthetically created, spanning the pose parameters listed in Table 1. This design aimed to encompass the views expected during flight testing. The same .obj file was used for the drogue in both EO and LWIR, but the pose parameters differ significantly in Table 1 because the object is being seen from different angles for each dataset. The imagery used for LWIR training was converted to grayscale before saving as a 3-channel .png to better resemble LWIR imagery. The LWIR and EO imagery was sized at



**Figure 2.** Blender 4.0 Render of Learjet 25D Model.

1024 × 768 and 864 × 864 pixels, respectively, with horizontal fields of view of 40° and 49° to match the expected camera imagery from the flight tests. Although the real EO camera produced images at 2848 × 2848 pixels, the imagery was scaled down to 864 × 864 pixels to reduce processing time. Examples of the training imagery can be seen in **Figure 3**.

Training-time data augmentation was also used according to Choate et al.<sup>[11]</sup> varying parameters such as translation, mixup, hue saturation, and value, but not altering scale. For the drogue data, this training-time data augmentation disabled rotational variation to the imagery. This was due to the symmetry of the drogue and special care taken to ensure rotational invariance of the components, which would be hindered if training-time augmentation rotated the image.

A new model was trained for each dataset, each starting from the YOLOv5s model, utilizing pretrained weights from the Ultralytics team.<sup>[3]</sup> This network was chosen due to its speed and accuracy. The number of epochs was 1500, patience was 30, and batch size was 32. In each case patience triggered, stopping training at epochs 662, 189, and 173 respectively for the LWIR Learjet, LWIR drogue, and EO drogue. These were trained on desktop PCs with the following hardware: AMD 7980X CPU, 128 GB DDR5 6400 MHz RAM, a RTX 4090 GPU; which took ≈20, 5, and 3 h to train each respective network.

### 3.4. Pseudo-Labeling Real and Infrared Images

In the 4th step of the process, there are three sub-steps to pseudo-labeling the real flight test images: 4) Pseudo-label real images 4a) CNN infers bboxes 4b) Compute 6DoF pose via Solve-PnP 4c) Label: Project points to image 4d) Tune and repeat.

For Step 4a, an image is passed to the CNN for inference. To filter outputs of YOLOv5 there are three relevant parameters:

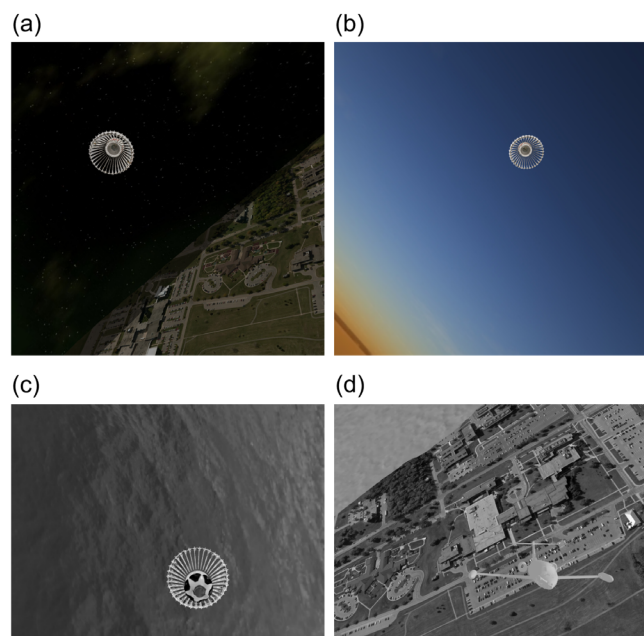
objectness confidence, class confidence, and non-max suppression (NMS) on intersection over union (IOU). Each of these parameters can be tuned to balance the number of predictions generated by the CNN and the number of false positives. Some false positives are acceptable due to Solve-PnP with RANSAC's ability to perform in the presence of outliers. The specific values utilized are detailed in a **Table 2**, 3.5, along with accompanying discussion. In this study, duplicate class ID predictions with lower objectness confidence will be discarded, as there should only be one of each component in a given image. Inference was accomplished via CUDA 12.1 with an ONNX model via OpenCV C++ API.<sup>[77]</sup>

For Step 4b, the centers of predicted bboxes are paired with the a priori knowledge of the 3D point cloud center for each component. These pairs, along with the camera intrinsic values, are passed to Solve-PnP with RANSAC<sup>[78]</sup> to obtain a 6DoF pose estimate for the object. The Solve-PnP algorithm requires at least four pairs to make an estimate, but tends to perform better with at least five pairs because RANSAC has the ability to discard outlying pairs and minimize error.

Step 4c uses the same process as the synthetic imagery generation discussed in Section 3.2. The 6DoF pose is used to project the point clouds of each component to an image, fit bboxes around, and correct for perspective distortion. Thus creating labels for the real EO and real infrared images.

Finally, Step 4d refers to tuning and repeating this process; this is because a user should identify good objectedness, class, and NMS IOU thresholds to appropriately restrict the number of false positives while allowing enough true positives for Solve-PnP to function well. In general, the Solve-PnP algorithm functions best with more points, even if there is noise or false positives within those points. Conversely, given too few points, Solve-PnP can provide poor estimates or be unable to use RANSAC to filter the false positives. This is why one should execute the entire process on their dataset before assessing if they should repeat it.

The authors herein utilized Python to make videos of the flight tests with YOLO predictions and pseudo-labeled images side-by-side for different objectness values. This allowed balancing of false positives from YOLO and the number of images which had 6DoF poses predicted. The values used are displayed in Table 2. Only the objectness confidence was modified; the remaining values used follow best practices from the Ultralytics team, the creators of YOLOv5, and Choate et al.<sup>[11]</sup> This still allowed bad pose estimates through, hence further filtering was done before determining which pseudo-labeled images could be used for further training, which will be discussed in Section 3.5.



**Figure 3.** Synthetic Training Imagery Examples. a) EO Droogie, b) EO Droogie, c) Grayscale Droogie, and d) Grayscale Learjet.

**Table 2.** Objectness, class confidence, and NMS values.

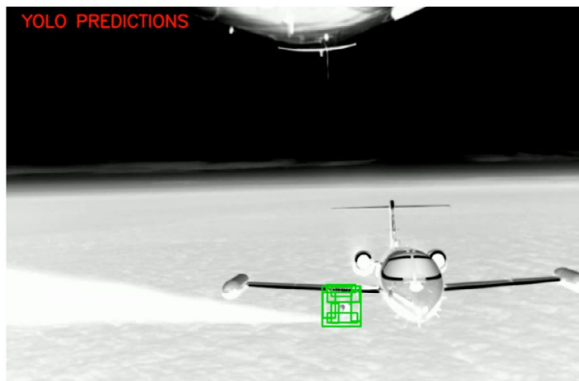
Dataset	Objectness	Class	NMS
LWIR Learjet	0.85	0.5	0.45
LWIR Droogie	0.15	0.5	0.45
EO Droogie	0.15	0.5	0.45

### 3.4.1. Example Pseudo-Labeled Images

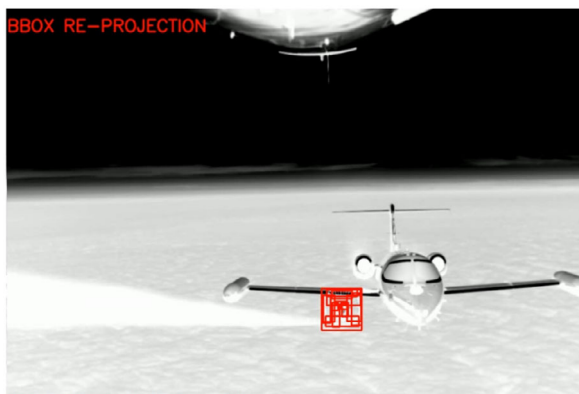
**Figure 4** illustrates an example of Step 4a, in which the YOLO CNN labeled bboxes on the image. Subsequently, **Figure 5** demonstrates an example of Step 4c, in which the 6DoF pose from Solve-PnP was utilized to project the bboxes back to the image. A key observation here is that Step 4b only requires four bboxes from Step 4a to pseudo-label an image with all the desired potential boxes or features.

### 3.4.2. Camera Intrinsic and Calibration

In computer vision, any computation involving projective geometry is dependent on the estimated camera intrinsics, making the accuracy of any method dependent on the camera intrinsic estimate accuracy. While an inaccurate camera calibration will result in incorrect 6DoF pose estimates, it may not necessarily result in poorly pseudo-labeled images. The camera intrinsics in this setup were calculated with two methods. For the LWIR camera, no chessboard calibration was performed during the event, hence a perfect calibration without distortion was assumed. Equation (1) shows how to calculate this.  $f_x$  and  $f_y$  refer to the focal length in pixels,  $C_x$  and  $C_y$  refer to the coordinate of the center pixel for the calibration, and  $hFoV$  refers to the horizontal field of view in radians.



**Figure 4.** LWIR Drogue Day Flight Left Cam Original BBox Labels.



**Figure 5.** LWIR Drogue Day Flight Left Cam Pseudo-Labeled.

$$f_x = f_y = \frac{C_x}{\tan\left(\frac{hFoV}{2}\right)} \quad (1)$$

The EO images underwent preprocessing to correct for distortion and resize them from  $2848 \times 2848$  to  $864 \times 864$  pixels. Once resized, images captured of a chessboard were used to execute Zhang's<sup>[79]</sup> method of camera calibration using OpenCV. Then the distortion parameters were used with OpenCV to undistort the images. When undistorted, the distortion parameters became obsolete, thus all distortion parameters received by the Solve-PnP algorithm were set to 0. This process yielded calibration parameters seen in **Table 3** and labeled as "EO".

Furthermore, **Table 3** presents the EO camera intrinsics calculated from the checkerboard calibration and the LWIR camera intrinsics calculated via Equation (1). These "EO" intrinsics were utilized as a base case in an experiment aimed at showcasing the negligible impact of intrinsics on pseudo-labeling images with our approach. It is important to note that varying the camera FoV can significantly affect the 6DoF pose estimates by the Solve-PnP algorithm. However, the experiment illustrates that if the same intrinsics used for Solve-PnP are applied to label the image, the resulting bboxes exhibit negligible differences versus this process being completed with a different camera calibration.

This experiment utilized the CNN trained on purely synthetic EO imagery to make bbox predictions on the real daytime flight test imagery. These bboxes were then used by Solve-PnP with a camera calibration to calculate pose, position and orientation, of the drogue. Subsequently, the components of the drogue for those poses were reprojected onto the image for each set of intrinsics to create bboxes. This process resulted in multiple datasets of bboxes, that could be compared to one another to measure the impact of intrinsic calibration. The EO calibration was used as the base case for comparison whereas all other calibration's sets of bboxes were compared to that. The other calibrations utilized for comparison were calculated in two ways: first was to vary the focal length from the base case by 5% and second was to choose the same  $C_x$  and  $C_y$  as the EO calibration or the assumed perfect values,  $C_x = C_y = 432$  for this case. After applying filtering techniques for bad pose estimates, the comparison focused on examining the consistency of 2D bbox centers between the reprojections. This filtering was necessary because, even though the same bboxes were used, the difference in intrinsics could cause Solve-PnP to find a bad or a good pose that would have otherwise been the opposite.

It is crucial to consider the distortion parameters used in this process and implementation should strive to set them to 0. Our work accomplished this by preprocessing the images to undistort them and only ever working with or saving images that had been undistorted. The significance of distortion parameters stems

**Table 3.** Camera intrinsics.

Camera	$f_x$	$f_y$	$C_x$	$C_y$
LWIR	1406.7	1406.7	512.0	384.0
EO	963.9	963.8	535.4	489.7

from the fact that a user can supply Solve-PnP with both distortion parameters and distorted images to obtain accurate 6DoF pose estimates. Users seeking 6DoF precision and streamlined processing in 6DoF pose estimation should consider this approach. However, it's important to acknowledge that reprojection of 3D points onto a 2D distorted image while properly compensating for distortion parameters is non-trivial and left as an exercise for the reader.

### 3.5. Filtering Pseudo-Labeled Images

Only using the various confidence values mentioned in Section 3.4 for filtering bad predictions is not recommended. This is because being too restrictive on the CNN prevents Solve-PnP from performing well; the more points Solve-PnP receives the better, even if some portion may be false positives or imprecise. However, allowing too many predictions from the CNN can be undesirable because too many false positives will enable Solve-PnP pose estimates when the object of interest is not in the scene.

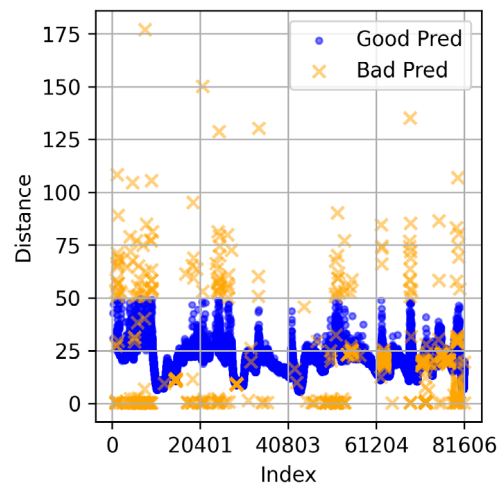
This is of special concern to transfer learning to real scenarios because the CNN has never seen the scenarios or features it is supposed to avoid predicting on, which could look very similar to the object in question. This was observed in initial testing where a circular logo was mistakenly predicted as the drogue basket when the confidence was tuned too low. However, the confidence values were able to be tuned to allow some false positives through, without allowing enough through to make pose predictions on those features. Another concept affecting the tuning is the receptive field of the CNN; it was observed during these efforts that the false positives reduced when the object of interest was in the image, that is, an image with a couple false positives on similar-looking features but no object of interest would have fewer false positives if the object was in the image.

Even after tuning of the CNN parameters, there were still images with poor pose estimates. To address this, simple filtering of the pseudo-labeled images was performed to discard the poorly pseudo-labeled images. Typically, Solve-PnP did not perform well when provided with only four pairs, so those estimates were discarded as part of the pseudo-labeling process. Additionally, images with estimates outside the expected distance from the camera were discarded; for example, during the flight test, the drogue was never closer than 5 m from the EO camera, nor further away from the LWIR camera than the length of the hose to which it was attached. Finally, an average pixel error was calculated by comparing predicted bbox centers by YOLO in the previous Step 4a against the center of the same bbox class after it was reprojected back to the image in Step 4c. These filtering criteria are summarized in Table 4.

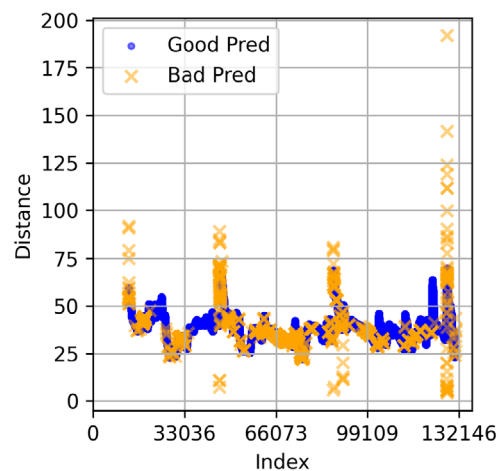
Figure 6, 7, and 8 illustrate the images classified as good (blue) and bad (orange) for each dataset used for the retraining outlined in Section 3.6. If the figures are considered independently, a reader might mistakenly conclude that the filters removed an excessive number of images, resulting in too few remaining for training. Therefore, Table 4 includes a column indicating the significant number of images that remained after applying the filter and conducting an 80/20 training-validation split. These figures demonstrate that simple filters can remove all

**Table 4.** Pseudo-Labeled Datasets' filtering Criteria.

Object	Minimum confidence	Minimum # predictions	Distance [m]	Maximum avg pixel error	Training image count
LWIR Learjet	0.85	5	(15, 70)	5	92 446
LWIR Drogue	0.15	5	(2.2, 26)	5	89 225
EO Drogue	0.15	5	(3, 50)	5	57 118



**Figure 6.** EO Day Flight Filtered Images.

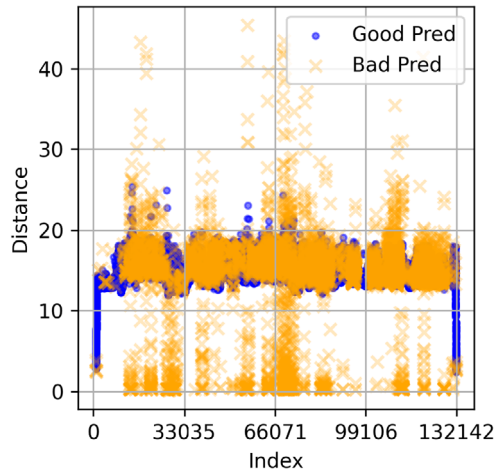


**Figure 7.** LWIR Learjet Day Flight Left Cam Filtered Images.

the bad predictions, although they may also remove some good predictions; it is up to a user to tune their filtering to their application.

As a reminder to the reader, the imagery from Flight 1 was broken into training and validation datasets via 80/20 split; the split was only the "left" camera for the LWIR data. Then that data had the filters mentioned here applied to it. The filters applied are summarized in Table 4, validation image count is omitted from the table.





**Figure 8.** LWIR Drogue Day Flight Left Cam Filtered Images.

### 3.6. Retraining the CNN

Each of the entries in Table 4 resulted in a new dataset of images and bbox labels. These images were combined with the synthetic training and validation images originally used to train the networks. Each network was then trained for 100 epochs on their new datasets, with a patience setting of 20 epochs. This took  $\approx 12$ , 14, and 23 h for the EO, LWIR Learjet, and LWIR drogue networks to train, respectively. All networks completed the 100 epochs without early stopping.

An additional network was trained for more than 800 epochs and compared to the network trained for 100 epochs. While longer training showed slight improvements in the typical CNN metrics like mean Average Precision (mAP), it did not show clear improvements for the metrics being assessed by this work, as discussed in Section 3.7. Therefore, 100 epochs were considered sufficient to demonstrate the benefits of this transfer learning pseudo-labeling technique.

### 3.7. Measuring Improvements

Multiple parameters were examined to show the improvements of this technique, both qualitatively and quantitatively. The metrics reported are good pose counts, bad pose counts, average prediction count, and average 2D reprojection error. The good pose count is defined as the number of images in which Solve-PnP received enough bbox pairs and produced a pose which was considered good. The bad pose count was when a pose was received from Solve-PnP, but it was considered bad, that is, outside the expected range or with a large reprojection error. A pose is considered good or bad per the filtering criteria in Table 5 and further discussed in Section 4. Table 5's filtering criteria is an update to Table 4's criteria, but slightly more strict for reporting on the test data.

The average prediction count is the number of YOLO predictions made per image across an entire dataset. The average 2D reprojection error is calculated as the mean pixel distance between the YOLO predictions and the reprojections. This distance is first averaged over each image and then averaged across

**Table 5.** Filtering Criteria for Determining good versus bad poses.

Object	Minimum confidence	Minimum # predictions	Distance [m]	Maximum avg pixel error
Lear_15	0.15	5	(15, 70)	5
Lear_85	0.85	5	(15, 70)	5
Lear_rtrn	0.15	5	(15, 70)	5
Drogue	0.15	5	(1, 20)	5
Drogue_rtrn	0.15	5	(1, 20)	5
EO_Drogue	0.15	5	(3, 50)	5
EO_Drogue_rtrn	0.15	5	(3, 50)	5
EO_Drogue_MoCap	0.15	5	(3, 50)	5

all images that had a predicted pose. Plots of the distance to the object from the camera are used to show a qualitative assessment of the various networks.

The pseudo-labeled data used to further train the models was all from Flight 1; for the LWIR cameras, only the left was used for the training data. Metrics are presented for each network against both the data used as the train validation split, as well as an additional flight and cameras during the evening of that same day. For the LWIR data, metrics are shown for those networks against both flights and both left and right cameras. This is to show performance on completely unseen data by the networks and the improvements made by the additional training.

### 3.8. Motion Capture Trained Comparison

The work herein was also compared to another CNN trained with the use of a MoCap room. The details for this work are in Worth's work.<sup>[41]</sup> This network was trained with a similar approach, however a MoCap room and AprilTags were used to get the truth pose of objects in the images, allowing for very precise labeling of bboxes in real imagery. The MoCap technique also showed promising results of transfer learning to real flight imagery; however, this network was trained on and only analyzed against EO imagery of the drogue. Hence, this network was compared to the same networks as the EO networks trained herein.

## 4. Results

This section presents the outcomes of the experiments, organized into three key areas. Firstly, we examine how camera intrinsic parameters are not essential in Section 4.1. Subsequently, we delve into the EO analysis of retraining (Section 4.2) as well as show how our method is comparable to a system built with a highly precise MoCap room. This is followed by a comprehensive exploration of the LWIR analysis of retraining (Section 4.3).

The filtering criteria used to pseudo-label the datasets earlier in Table 4 was modified for the results section to provide a slightly more stringent set of bounds for the LWIR drogue dataset. This can be seen in Table 5. The tables and plots in this section use these criteria to determine if a pose estimate was good or

bad. The average pixel error and the prediction count averages are calculated independently of the filtering criteria.

Note, for all retrained networks, the same objectness, class confidence, and NMS values were used; 0.15, 0.5, and 0.45, respectively. This is noteworthy due to the fact the labeling of the Learjet utilized 0.85 objectness confidence to best filter its estimates.

#### 4.1. Non-Essential Camera Intrinsic

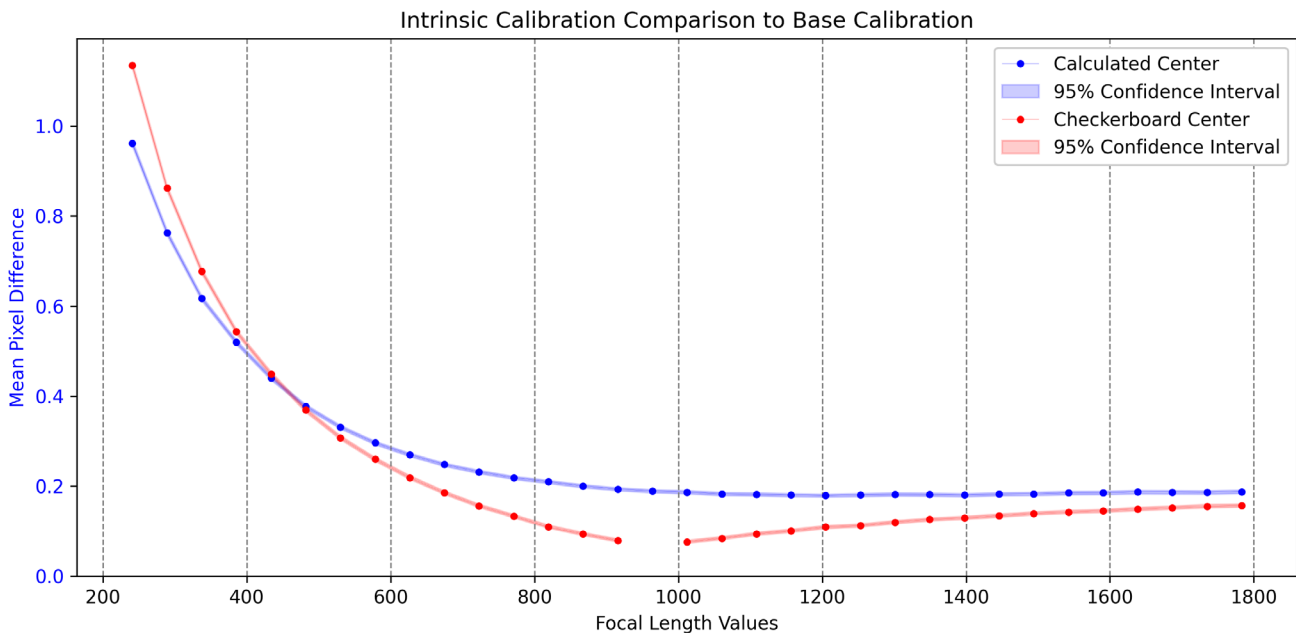
For the experiment described in Section 3.4.2, the results are summarized in **Figure 9**. The pseudo-labeled bboxes found from the checkerboard calibration intrinsics were used as the base case for comparison. These can be found in Table 3. Those bboxes were then compared to pseudo-labeled bboxes from the system using various other sets of intrinsics. The other calibrations utilized for comparison were calculated in two ways: first was to vary the focal length from the base case by 5%, and second was to choose the same  $C_x$  and  $C_y$  as the base case (denoted as red), or the assumed perfect center values of  $C_x = C_y = 432$  (denoted as blue).

This test demonstrated that the effect of poor or incorrect intrinsic parameters is generally negligible for pseudo-labeled images, in our process. The most significant effects occurred when the intrinsic values deviated substantially from the original, resulting in a lens with a very wide FoV. Such poor calibration is unrealistic in practical scenarios. Figure 9 also reveals that the  $C_x$  and  $C_y$  values seemed to be the primary cause of the difference for the majority of the experiment, overshadowing the effect of changes in focal length for reasonable focal length margins. Overall, since the difference between the base case and reasonable camera calibrations was under a pixel, it was concluded that the effect of the calibration is negligible for the pseudo-labeling process herein.

It is worth noting that a reader should consider their specific camera and object setup in relation to these results. The drogue observed in the images for the test in Figure 9 was positioned far enough from the camera that the effects of perspective distortion were minimal. Therefore, users with objects significantly closer to the camera or significantly larger may experience higher levels of perspective distortion, which could change the significance of their camera calibration in this pseudo-labeling process. For typical use cases where objects do not exhibit significant distortion, the intrinsic parameters are likely less critical. These findings underscore the robustness of the proposed pseudo-labeling approach to variations in camera calibration, further validating its efficacy for everyday users without expensive equipment or truth data using it for object detection labeling.

#### 4.2. EO Analysis of Retraining

The first look at the results for this process can be seen via a comparison of the EO camera mounted in the Learjet cockpit predicting the drogue position. **Table 6** and **7** show that there is a slight improvement across all parameters when continuing the training of a CNN with real images labeled with the proposed technique. Specifically, Table 6 demonstrates that the retrained network (EO\_Drogue\_rtrn) exhibits improvements compared to the original network (EO\_Drogue) in both datasets (Day and Nt). The results for these two networks demonstrated statistical significance in average pixel error and average prediction count as determined by the Wilcoxon Rank-Sum test with an alpha of 0.01;  $P$ -values were significantly less than 0.001. Similarly, the results in Table 7 showed statistical significance for both datasets as determined by the Chi-Square test conducted on the good and bad counts using an alpha of 0.01.  $P$ -values were significantly less than 0.001. These findings underscore the efficacy of the



**Figure 9.** Intrinsic Calibration Effects on Pseudo-Labeling.

**Table 6.** 2D EO performance  $\left( \frac{\text{Avg Pixel Error}}{\text{Prediction Counts}} \right)$ .

Network	Day	Nt
EO_Drogue	0.96 7.22	1.90 10.76
EO_Drogue_rtrn	<b>0.57</b> <b>7.51</b>	<b>1.03</b> <b>10.89</b>
EO_Drogue_MoCap	1.54 13.50	1.06 35.06

**Table 7.** 3D EO performance  $\left( \text{estimates } \frac{\text{Good}}{\text{Bad}} \right)$ .

Network	Day	Nt
EO_Drogue	71,417 619	50,496 511
EO_Drogue_rtrn	<b>72,800</b> <b>132</b>	<b>54,062</b> <b>153</b>
EO_Drogue_MoCap	40,329 209	53,243 184

pseudo-labeling and retraining approach, affirming its status as the most effective method evaluated.

Additionally, these tables show that the results are comparable or better than a network trained with a 3D MoCap room and real imagery. The average bbox prediction count for the MoCap network should not be compared to the other networks because it was trained with a different set of features on the drogue. The MoCap network included some 65 features versus 13 possible features for this work. Our network outperformed the MoCap network during the day flight; this was attributed to the training data for the MoCap network having been collected in a large room without perfect lighting. Conversely, the MoCap network outperformed our original network, (EO\_Drogue) for the night portion of flight, likely for this same reason. However, after being retrained, our approach performed better. When comparing the retrained network versus the MoCap network the average pixel error demonstrated statistical significance via Wilcoxon Rank-Sum test with similar parameters as before. Similarly, the results in Table 7 showed statistical significance for the MoCap and retrained networks demonstrated by Chi-Square test for the day flight with similar parameters as before, but the night flight did not show statistically significant differences.

Overall, these results show that our approach to initial training has very good transfer learning unto itself for EO imagery, that retraining the network can have a positive effect, and that this approach can outperform networks trained with real data from a MoCap room.

### 4.3. LWIR Analysis of Retraining

Analysis of the results from the LWIR imagery truly highlights the effectiveness of our pseudo-labeling approach. **Table 8** reveals that the average pixel error was higher for all of the purely synthetically trained models on LWIR images compared to the EO results from Section 4.2, Table 6. This discrepancy is understandable, considering the significant leap required for transfer

**Table 8.** 2D LWIR performance  $\left( \frac{\text{Avg Pixel Error}}{\text{Prediction Counts}} \right)$ .

Network	L_Day	R_Day	L_Nt	R_Nt
Lear_15	5.21 20.29	12.45 17.99	8.15 18.57	11.81 12.12
Lear_85	1.84 14.93	6.54 12.00	2.96 12.28	3.11 8.97
Lear_rtrn	<b>0.29</b> <b>23.21</b>	<b>0.58</b> <b>20.95</b>	<b>1.09</b> <b>22.50</b>	<b>1.15</b> <b>18.62</b>
Drogue	2.63 8.56	3.84 7.79	2.21 7.39	8.86 5.96
Drogue_rtrn	<b>0.36</b> <b>11.28</b>	<b>0.37</b> <b>11.36</b>	<b>0.47</b> <b>11.37</b>	<b>0.64</b> <b>10.64</b>

learning on LWIR imagery compared to real EO images in this study. The other metrics are not useful to compare directly between the EO and LWIR experiments because the numbers of images each had was very different and the components identified on the objects were different. Even for the drogue, distinct components were defined for the front (EO) and back (LWIR) detection.

The Learjet appears multiple times in **Table 8** and **9**, represented by “Lear\_15” and “Lear\_85” entries. These denote the Learjet network trained exclusively on synthetic data; however, it was run against the data twice with different objectedness values, 0.15 and 0.85. This was to allow the reader to observe the same values used to make the pseudo-labels, Table 4, and the values used by the retrained network, fully summarized in Table 5. It can also be seen that a more lenient objectedness value had a reduced bad pose estimate count with the retrained network versus the stricter objectedness prior to pseudo-label training, seen in Table 9.

These entries also serve to illustrate that a higher number of predicted bboxes does not necessarily equate to better pose predictions. For instance, “Lear\_15” predicts more bboxes on average than “Lear\_85” for each dataset, yet it also yields fewer accurate pose estimates, as shown in Table 9. Therefore, this underscores the importance of tuning the parameters of a CNN before employing its results for the proposed pseudo-labeling technique and the dataset against which it is employed.

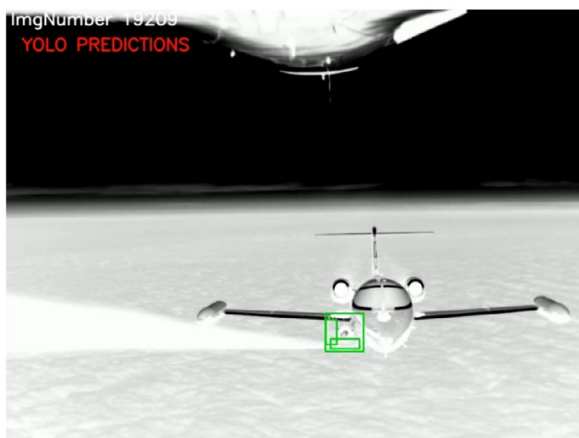
The LWIR retrained networks demonstrated significant improvement across all of our key metrics, seen in Table 8 and 9. First, the average pixel error decreased to under or near

**Table 9.** 3D LWIR performance  $\left( \text{estimates } \frac{\text{Good}}{\text{Bad}} \right)$ .

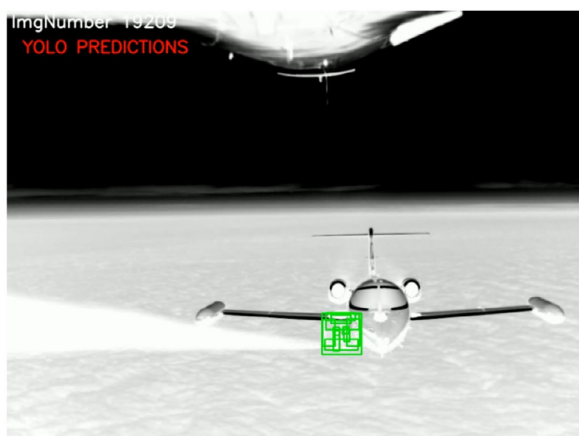
Network	L_Day	R_Day	L_Nt	R_Nt
Lear_15	107,353 15,459	82,368 37,573	16,313 9,630	16,153 13,583
Lear_85	105,560 1,614	101,059 15,484	20,058 2,879	21,136 1,608
Lear_rtrn	<b>117,573</b> <b>85</b>	<b>117,407</b> <b>191</b>	<b>23,036</b> <b>147</b>	<b>22,998</b> <b>100</b>
Drogue	111,619 9,847	77,520 5,225	31,524 1,288	5,760 1,002
Drogue_rtrn	<b>130,662</b> <b>84</b>	<b>130,632</b> <b>55</b>	<b>35,217</b> <b>6</b>	<b>34,642</b> <b>41</b>

a single pixel, a 2.7–13.8x decrease between each of the datasets, indicating the retraining properly taught the networks the LWIR appearance of the objects while remaining precise. Additionally, the retrained models produced higher bbox prediction counts at the same objectness values. This increase in bbox predictions was complemented by a rise in the number of good 6DoF pose estimates and decrease in bad estimates, showcasing the overarching system's increased robustness and reliability on these data sets.

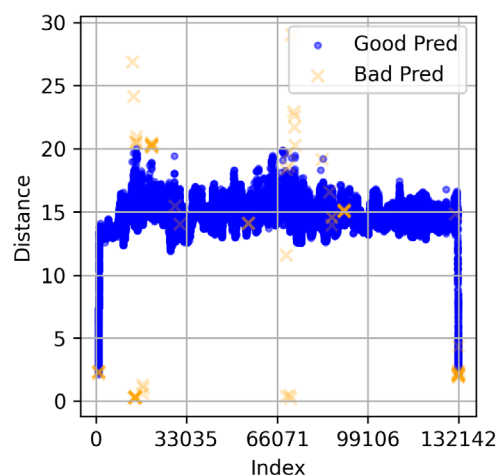
Statistical significance in the results was demonstrated by comparing the retrained networks to their original counterparts using various tests. Specifically, Table 8 shows that the retrained networks exhibit statistically significant improvements in both average pixel error and prediction counts over their original versions, as confirmed again by the Wilcoxon Rank-Sum with an alpha of 0.01 and *P*-values significantly less than 0.001. Similarly, Table 9 indicates that the retrained network's performance outperforms the original networks, as confirmed again by the Chi-Square test on the good and bad counts using an alpha of



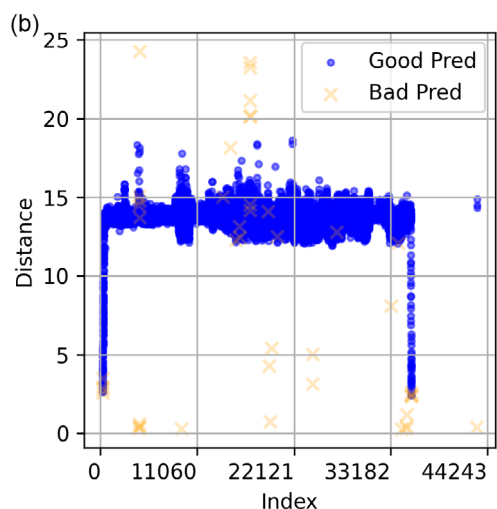
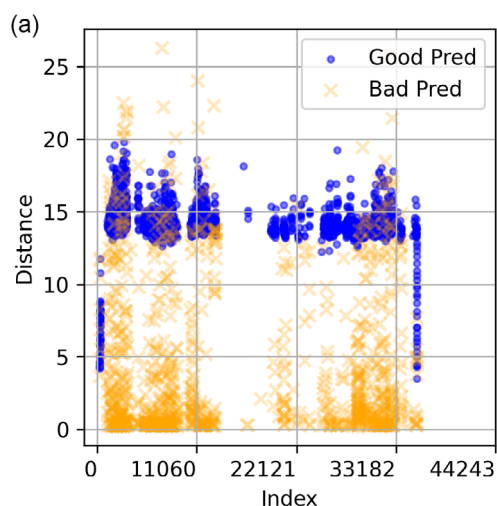
**Figure 10.** Example Image of Original Network Failing to Predict Enough BBoxes.



**Figure 11.** Example Image of Retrained Network Predicting Enough BBoxes.



**Figure 12.** LWIR Drogue Day Flight Left Cam Retrained Network.



**Figure 13.** Comparison of LWIR Drogue Night Flight Right Cam between a) Original and b) Retrained Networks. (a) LWIR Drogue Night Flight right Cam Original network, (b) LWIR Drogue Night Flight right Cam retrained network.



0.01 and  $P$ -values significantly less than 0.001. These findings emphasize the effectiveness of the retraining approach, highlighting it as the most successful method among those evaluated.

Figure 10 and 11 also show increased reliability and robustness, whereas these images show the network performing well in scenarios where it previously failed. The images illustrate that when the Learjet was behind the drogue, the original CNN would fail to make enough predictions for a pose estimate on the drogue. However, after retraining, it was no longer confused by that scenario; the figures show the same image of the dataset by the network before and after retraining.

Furthermore, Table 8 and 9 illustrate the increased number of good pose predictions and decreased number of bad pose predictions. The number of good pose predictions of the retrained network was greater than the total number of pose estimates by the original networks. Indicating that not only were the retrained networks producing better poses, but also had learned enough to estimate good poses on images the original networks were unable to predict poses on at all. This is better depicted in Figure 12b and 13a.

Figure 12 is the retrained network executing on the same data shown in Figure 8. It is evident that many of the bad pose estimates have been eliminated. Some of filtering techniques used to identify bad pose estimates may also be flagging acceptable values, showing a need for a deployed system to use more robust filtering.

Figure 13a shows that data was especially difficult due to being a night flight and the right camera being out of focus. However, Figure 13b shows that the retraining technique improved the ability of the system significantly, allowing for predictions where previously not possible, greatly reducing the number of bad pose estimates, and seeming to reduce the noise of the predictions that were considered good.

## 5. Conclusions and Future Work

In summary, this article details the steps necessary to implement a novel labeling approach for real imagery, without the need for camera intrinsics or 3D truth data. Experimental results demonstrated that precise camera calibration is not essential for this technique, even though precise intrinsics are necessary for accurate 6DoF poses. Our approach was validated using real EO and LWIR imagery, showing significant improvements in performance for the LWIR transfer learning. Moreover, our technique was comparable and even outperformed a similar system trained with highly precise sensors in a MoCap room.

This study is significant because it creates the ability to label real imagery for training neural networks. It enables the revival of archived real video and imagery of operational scenarios, making them useful for training CNNs. This directly addresses the critical need of robust computer vision-based automation systems for high quality, precise, and labeled imagery, as close to the desired deployment scenario as possible. Our labeling technique enables the training of neural networks on operational and application-specific imagery without requiring expensive sensor suites. Additionally, this approach allows for iterative training of

neural networks, continuously refining them as scenarios reveal performance limitations. This addresses the critical need for precisely labeled, high-quality data that computer vision community is starved for, as it is applied to autonomous navigation, robotics, and the creation of more sophisticated and reliable artificial intelligence systems.

There are obvious limitations to this work. The first limitation is the lack of 6DoF truth data to assess the resulting system accurately. A clear direction for future work would be to evaluate the overall system's ability to make 6DoF pose estimates with highly precise truth data.

Another limitation lies in the filtering techniques used. The techniques employed in this study were deliberately simple to demonstrate the robustness of the approach and its general applicability to various datasets. However, most real-world data includes a time series aspect, presenting an opportunity for future work to implement algorithms like Kalman filters<sup>[80]</sup> or more advanced filtering methods. These advanced techniques could be applied in both 2D and 3D to reduce noise, false positives, false negatives, and bad pose estimates with which our current filtering method contended. Additionally, the reprojection error used as a filtering technique could be assessed for usage in a deployed system as a method to detect for bad Solve-PnP estimates.

Furthermore, this study only applied the technique once, showing significant benefits even after a single iteration. Future research could apply this approach to multiple disparate datasets to demonstrate that the pseudo-label trained network could be used to label other imagery that was previously unable to be labeled and to fine tune deployed networks when situations are found where they are deficient. Finally, future work could also focus on automation of the entire process. This could create more opportunities for application with other austere sensors and in diverse environmental conditions.

In conclusion, this study presents a novel technique for creating labeled imagery from real-world imagery, leveraging transfer learning from synthetic images. Our method demonstrated the effectiveness of further training CNNs on real imagery labeled by our technique, addressing a critical need in the computer vision community for high-quality, precisely labeled data from actual operational environments. Notably, our approach does not require well-calibrated camera intrinsics, which is impactful because users may not have intrinsic parameters for their existing datasets, especially those captured with multiple cameras or older data. The significant improvements observed in performance, particularly in LWIR transfer learning, underscore the potential of this approach to enhance autonomous navigation, robotics, and artificial intelligence systems. This work not only contributes to current methodologies but also opens avenues for future research to refine and expand its application across diverse datasets and scenarios.

## Code Availability

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

## 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.

## Acknowledgements

The authors would like to thank all our sponsors, including the Naval Air Systems Command (USN/NAVAIR) and Aerospace Systems Directorate (USAF/AFRL/RQ). Your partnership and financial support made this research possible.

## Conflict of Interest

The authors declare no conflict of interest.

## Author Contributions

**Jeffrey Choate:** writing—original draft: (lead); writing—review & editing: (lead). **Derek Worth:** writing—review & editing: (supporting). **Scott L Nykl:** conceptualization: (supporting); data curation: (supporting); writing—review & editing: (supporting). **Clark Taylor:** writing—review & editing: (supporting). **Brett Borghetti:** writing—review & editing: (supporting). **Christine Schubert Kabban:** writing—review & editing: (supporting). **Ryan Raettig:** writing—review & editing: (supporting).

## Data Availability Statement

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

## Keywords

convolutional neural network, Perspective-N-Point algorithm, pseudo-labeling, real time pose estimation, synthetic imagery generation, transfer learning, YOLOv5 object detection

Received: July 10, 2024  
Revised: December 12, 2024  
Published online:

- [1] J. D. Anderson, S. Nykl, T. Wischgoll, in *Advances in Visual Computing: 14th Int. Symp. on Visual Computing, ISVC 2019, Lake Tahoe, NV, USA, October 7–9, 2019, Proc., Part II 14*, Springer, New York, NY **2019**, pp. 154–165.
- [2] S. Nykl, C. Mourning, M. Leitch, D. Chelberg, T. Franklin, C. Liu, in *38th Annual Frontiers in Education Conf.*, IEEE, Piscataway, NJ **2008**, pp. F3B–21.
- [3] G. Jocher, A. Stoken, J. Borovec, NanoCode012, C. Stan, L. Changyu, Laughing, tkianai, Adam Hogan, lorenzomammanna, 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**.
- [4] B. M. Haralick, C.-N. Lee, K. Ottenberg, M. Nölle, *Int. J. Comput. Vis.* **1994**, 13, 331.
- [5] J. Choate, D. Worth, S. Nykl, C. Taylor, B. Borghetti, C. S. Kabban, R. Raettig, *From Simulation to Reality: Transfer Learning for Automating Pseudo-Labeling of Real and Infrared Imagery*, YouTube Video, **2024**, <https://youtu.be/Amc1vbc10Jg>.
- [6] N. Clark, *What it Takes to be a Boom Operator* **2015**, <https://www.af.mil/News/Article-Display/Article/585353/what-it-takes-to-be-a-boom-operator>, (accessed: June 2024).
- [7] J. Hansen, G. Romrell, N. Nabaa, R. Andersen, L. Myers, J. McCormick, in *Proc. of the 19th Int. Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS 2006)* **2006**, pp. 674–685.
- [8] K. Schweikhard, Results of NASA/DARPA Automatic Probe and Drogue Refueling Flight Test, Technical Report **2008**.
- [9] J. D. Anderson, R. M. Raettig, J. Larson, S. L. Nykl, C. N. Taylor, T. Wischgoll, *Mach. Vis. Appl.* **2022**, 33, 31.
- [10] C. Parsons, S. Nykl, in *International Symposium on Visual Computing*, Springer, New York, NY **2016**, pp. 605–615.
- [11] J. Choate, D. Worth, S. Nykl, C. Taylor, B. Borghetti, C. S. Kabban, *Neural Comput. Appl.* **2024**, 36, 1261.
- [12] J. C. Lynch, *Monocular Pose Estimation for Automated Aerial Refueling via Perspective-n-Point. Technical Report*, Air Force Institute of Technology, Wright-Patterson AFB OH **2022**.
- [13] Q. Tran, J. Choate, C. N. Taylor, S. Nykl, D. Curtis, in *IEEE/ION Position, Location and Navigation Symp. (PLANS)*, IEEE, Piscataway, NJ **2023**, pp. 128–136.
- [14] D. Worth, J. Choate, J. Lynch, S. Nykl, C. Taylor, *Neural Comput. Appl.* **2024**, 36, 10143.
- [15] Amazon Mechanical Turk, <https://www.mturk.com> (accessed: June 2024).
- [16] B. Dwyer, J. Nelson, T. Hansen, et al. Roboflow (Version 1.0) **2024**, <https://roboflow.com>, (accessed: January 2024).
- [17] Amazon Web Services, Amazon Sagemaker, <https://aws.amazon.com/sagemaker/> **2023**, (accessed: May 2024).
- [18] B. C. Russell, A. Torralba, K. P. Murphy, W. T. Freeman, *Int. J. Comput. Vis.* **2008**, 77, 157.
- [19] Labellerr Inc., Labellerr **2024**, Computer Software, <https://www.labellerr.com/>, (accessed: June 2024).
- [20] T. Hammarqvist, *Automatic Annotation of Models for Object Classification in Real Time Object Detection*, **2021**.
- [21] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, in *IEEE Conf. on Computer Vision and Pattern Recognition*, IEEE, Piscataway, NJ **2009**, pp. 248–255.
- [22] B. Pande, K. Padamwar, S. Bhattacharya, S. Roshan, M. Bhamare, in *Int. Conf. on Applied Artificial Intelligence and Computing (ICAIC)*, IEEE, Piscataway, NJ **2022**, pp. 976–982.
- [23] E. Real, J. Shlens, S. Mazzocchi, X. Pan, V. Vanhoucke, in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, IEEE, Piscataway, NJ **2017**, pp. 5296–5305.
- [24] Blender Online Community, *Blender – A 3D Modelling and Rendering Package*, Blender Foundation, Amsterdam **2018**, <http://www.blender.org>, (accessed: January 2024).
- [25] E. Games, Unreal Engine (version 5.0) **2024**, <https://www.unrealengine.com>, (accessed: January 2024).
- [26] A. Gil, A. Khurshid, J. Postal, T. Figueira, in *Anais Estendidos do XXXII Conf. on Graphics, Patterns and Images*, SBC, **2019**, pp. 237–242.
- [27] Unity Technologies, Unity Game Engine, Software, **2024**, <https://unity.com/>, (accessed: January 2024).
- [28] H. Hattori, V. N. Boddeti, K. M. Kitani, T. Kanade, in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, IEEE, Piscataway, NJ **2015**, pp. 3819–3827.
- [29] P. S. Rajpura, H. Bojinov, R. S. Hegde, *Object Detection Using Deep CNNs Trained on Synthetic Images*, arXiv:1706.06782, **2017**.

- [30] A. Rozantsev, V. Lepetit, P. Fua, *Comput. Vis. Image Understand.* **2015**, 137, 24.
- [31] J. Talukdar, S. Gupta, P. S. Rajpura, R. S. Hegde, in *5th Int. Conf. on Signal Processing and Integrated Networks (SPIN)*, IEEE, Piscataway, NJ **2018**, pp. 78–83.
- [32] Vicon.Vicon **2024**. <https://vicon.com/hardware/cameras/>, (accessed: June 2024).
- [33] Phasespace, Phasespace **2017**, <https://www.phasespace.com/x2e-motion-capture/>, (accessed: June 2024).
- [34] OptiTrack, Primex 41 **2024**, <https://optitrack.com/cameras/primex-41/>, (accessed: June 2024).
- [35] J. Hinchman, D. Schreiter, *Presentation to Aerial Refueling Systems Advisory Group (ARSAG)*, **2007**.
- [36] G. Balamurugan, J. Valarmathi, V. P. S. Naidu, in *Int. Conf. on Signal Processing, communication, Power and Embedded System (SCOPEs)*, IEEE, Piscataway, NJ **2016**, pp. 198–204.
- [37] R. C. Leishman, T. W. McLain, R. W. Beard, *J. Intell. Robot Syst.* **2014**, 74, 97.
- [38] X. Ma, Y. Meng, Y. Zhang, L. Bai, J. Hou, S. Yi, W. Ouyang, *An Empirical Study of Pseudo-Labeling for Image-Based 3D Object Detection*, arXiv:2208.07137, **2022**.
- [39] Zhenyu Li Zehui Chen, Shuo Wang, Dengpan Fu, Feng Zhao, in *Proc. of the IEEE/CVF Int. Conf. on Computer Vision*, IEEE, Piscataway, NJ **2023**, pp. 6929–6939.
- [40] J. Curro, J. Raquet, T. Pestak, J. Kresge, M. Smearcheck, in *Proc. of the 25th Int. Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS 2012)*, **2012**, pp. 774–782.
- [41] D. Worth, J. Choate, R. Raettig, S. Nykl, C. Taylor, *Neural Comput. Appl.*, **2024**.
- [42] T. Kiyokawa, K. Tomochika, J. Takamatsu, T. Ogasawara, *IEEE Robot. Autom. Lett.* **2019**, 4, 1972.
- [43] R. E. P. Ferreira, Y. J. Lee, J. R. R. Dórea, *Sci. Rep.* **2023**, 13, 13875.
- [44] E. Arazo, D. Ortego, P. Albert, N. E. O'Connor, K. McGuinness, in *International Joint Conference On Neural Networks (IJCNN)*, IEEE, Piscataway, NJ **2020**, pp. 1–8.
- [45] P. Cascante-Bonilla, F. Tan, Y. Qi, V. Ordonez, in *Proc. of the AAAI Conf. on Artificial Intelligence*, **2021**, Vol. 35, pp. 6912–6920.
- [46] N. Seedat, N. Huynh, F. Imrie, M. van der Schaar, *You Can't Handle the (Dirty) Truth: Data-Centric Insights Improve Pseudo-Labeling*, arXiv:2406.13733, **2024**.
- [47] A. Stathopoulos, G. Pavlakos, L. Han, D. N. Metaxas, in *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, IEEE, Piscataway, NJ **2023**, pp. 13092–13101.
- [48] J. Zhang, J. Li, X. Lin, W. Zhang, X. Tan, J. Han, E. Ding, J. Wang, G. Li, in *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, IEEE, Piscataway, NJ **2024**, pp. 16923–16932.
- [49] Z. Zhang, M. Chen, S. Xiao, L. Peng, H. Li, B. Lin, P. Li, W. Wang, B. Wu, D. Cai, in *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, IEEE, Piscataway, NJ **2024**, pp. 15291–15300.
- [50] H. Pham, Z. Dai, Q. Xie, Q. V. Le, in *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, IEEE, Piscataway, NJ **2021**, pp. 11557–11568.
- [51] J. Wu, D. Guo, G. Wang, Q. Yue, H. Yu, K. Li, S. Zhang, *IEEE Trans. Med. Imaging* **2024**, 43, 3098.
- [52] Y. Yang, K. J. Liang, L. Carin, *Object Detection as a Positive-Unlabeled Problem*, arXiv:2002.04672, **2020**.
- [53] J. Choi, M. Jeong, T. Kim, C. Kim, *Pseudo-Labeling Curriculum for Unsupervised Domain Adaptation*, arXiv:1908.00262, **2019**.
- [54] Y. Liu, W. Zhang, A. V. Vasilakos, L. Wang, arXiv:2404.06683, **2024**.
- [55] H. Kojima, N. Kaneko, S. Ito, K. Sumi, in *International Workshop on Frontiers of Computer Vision*, Springer, New York, NY **2022**, pp. 127–140.
- [56] G. Li, X. Li, Y. Wang, Y. Wu, D. Liang, S. Zhang, in *European Conference on Computer Vision*, Springer, New York, NY **2022**, pp. 457–472.
- [57] R. Cheng, S. Lucyszyn, *Sci. Rep.* **2024**, 14, 3150.
- [58] F. Dubourvieux, R. Audigier, A. Loesch, S. Ainouz, S. Canu, *Comput. Vis. Image Understand.* **2022**, 223, 103527.
- [59] J. Yang, S. Shi, Z. Wang, H. Li, X. Qi, in *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, IEEE, Piscataway, NJ **2021**, pp. 10368–10378.
- [60] S. R. Baek, J. Jang, *Expert Syst. Appl.* **2024**, 236, 121405.
- [61] J. Yu, L. Zhang, S. Du, H. Chang, K. Lu, Z. Zhang, Y. Yu, L. Wang, Q. Ling, in *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, IEEE, Piscataway, NJ **2022**, pp. 305–312.
- [62] L. Torrey, J. Shavlik, in *Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques*, IGI Global **2010**, pp. 242–264.
- [63] S. J. Pan, Q. Yang, *IEEE Trans. Knowl. Data Eng.* **2009**, 22, 1345.
- [64] Wikipedia Contributors, *Transfer Learning* **2024**, [https://en.wikipedia.org/wiki/Transfer\\_learning](https://en.wikipedia.org/wiki/Transfer_learning), (accessed: May 2024).
- [65] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, Q. He, *Proc. IEEE* **2020**, 109, 43.
- [66] F. E. Nowruzi, P. Kapoor, D. Kolhatkar, F. A. Hassanat, R. Laganieri, J. Rebut, *How Much Real Data Do We Actually Need: Analyzing Object Detection Performance Using Synthetic and Real Data*, arXiv:1907.07061, **2019**.
- [67] N. Yabuki, N. Nishimura, T. Fukuda, in *Advanced Computing Strategies for Engineering: 25th EG-ICE Int. Workshop 2018, Lausanne, Switzerland, June 10-13, 2018, Proc., Part I 25*, Springer, New York, NY **2018**, pp. 3–15.
- [68] M. Rad, V. Lepetit, in *Proc. of the IEEE Int. Conf. on Computer Vision*, IEEE, Piscataway, NJ **2017**, pp. 3828–3836.
- [69] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, N. Navab, in *Proc. of the IEEE Int. Conf. on Computer Vision, SSD 6D*, **2017**, pp. 1521–1529.
- [70] Y. Xiang, T. Schmidt, V. Narayanan, D. Fox, *POSECNN: A Convolutional Neural Network for 6d Object Pose Estimation in Cluttered Scenes*, arXiv:1711.00199, **2017**.
- [71] W.-L. Huang, C.-Y. Hung, I.-C. Lin, *IEEE Trans. Multimedia* **2021**, 24, 3025.
- [72] J. She, Y. Liu, R. Zhou, N. Qi, *CSAA/IET International Conference on Aircraft Utility Systems (AUS 2020)*, Online Conference **2021**, pp. 279–284. <https://doi.org/10.1049/icp.2021.0331>.
- [73] M. Sundermeyer, Z.-C. Marton, M. Durner, M. Brucker, R. Triebel, in *Proc. of the European Conf. on Computer Vision (ECCV)* **2018**, pp. 699–715.
- [74] B. Tekin, S. N. Sinha, P. Fua, in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, IEEE, Piscataway, NJ **2018**, pp. 292–301.
- [75] F. Wang, G. Wang, B. Lu, *Electronics* **2024**, 13, 1046.
- [76] Artec 3D, *Artec Leo 3D Scanner* **2023**, <https://www.artec3d.com/portable-3d-scanners/artec-leo>, (accessed: January 2024).
- [77] G. Bradski, *The OpenCV Library. Dr. Dobb's J. Softw. Tools* **2000**.
- [78] M. A. Fischler, R. C. Bolles, *Commun. ACM* **1981**, 24, 381.
- [79] Z. Zhang, *IEEE Trans. Pattern Anal. Mach. Intell.* **2000**, 22, 1330.
- [80] G. Welch, G. Bishop, et al., *Proc of SIGGRAPH, Course*, **1995**, Vol. 8, p. 41.