

Image Segmentation with Deep Neural Networks to Accelerate Long Range Pose Estimation in High Resolution Images for Automated Aerial Refueling

Andrew Lee^{*}, Will Dallmann[†], Scott Nykl[‡], Clark Taylor[§], and Brett Borghetti[¶]
Air Force Institute of Technology, WPAFB, OH 45433

Automated aerial refueling (AAR) provides unique challenges for computer vision systems. Aerial refueling maneuvers require high-precision, low-variance pose estimates. We quantify the performance of two stereo vision systems in a ground tests specially designed to mimic AAR. In this experiment, we achieve 3D pose estimation errors of 6cm on a target 30m from our vision system. Next, we propose a novel computer vision pipeline that leverages a convolutional neural network (CNN) to efficiently generate a 3D point cloud of the target object using stereo vision. Using our proposed approach, we are able to generate a high-fidelity 3D point cloud with ultra high-resolution imagery 11.3× faster than previous approaches.

I. Introduction

AUTOMATED aerial refueling (AAR) is an important emerging technology for the United States Air Force (USAF). Currently, remotely piloted aircraft (RPAs) cannot perform in-air refueling maneuvers due to the multi-second latency between the pilot and the remote aircraft. An AAR system bypasses this limitation by allowing the tanker to automatically control the receiver during the refueling process. The ability to refuel RPAs in flight would provide a valuable increase in operational endurance for their mission set. Additionally, an AAR system serves as an important intermediate step to a fully autonomous tanker. An AAR system needs the capability to control the tanker, its refueling boom, and the receiver for the duration of the aerial refueling procedure. Before control logic can be safely implemented, the system must have high precision relative pose estimation process that tracks the receiver in real time. In this paper, we implement a vision system that achieves such requirements.

The greatest challenge for an AAR computer vision system is the long range from the cameras on a tanker to the refueling contact point. The contact point is approximately 30m from the stereo cameras. At contact, the 3D pose estimation error must be less than 10cm. Due to aerial refueling mission constraints, we cannot modify existing aircraft or use global positioning system (GPS) signals to augment a vision-based approach. We have previously proposed a

^{*}2d Lt USAF, AFIT/ENG, 2950 Hobson Way, WPAFB, OH 45433

[†]Capt USAF, AFIT/ENG, 2950 Hobson Way, WPAFB, OH 45433

[‡]Assistant Professor Computer Science, AFIT/ENG, 2950 Hobson Way, WPAFB, OH, 45433

[§]Assistant Professor Computer Science, AFIT/ENG, 2950 Hobson Way, WPAFB, OH, 45433

[¶]Associate Professor Computer Science, AFIT/ENG, 2950 Hobson Way, WPAFB, OH, 45433

method for solving this problem using a stereo vision system [?]. Here, we seek to validate and improve that system.

One method for improving the accuracy of pose estimates is to increase the stereo images' resolution. Unfortunately, an increase in pixel count leads to a proportional increase in processing time. To mitigate this limitation, we draw inspiration from nature. The neural structure dedicated to processing imagery is too small to support high-resolution sensory over a wide field of view. Therefore, the biological eyes in predators have small, ultra-high-resolution fovea and a low-resolution periphery. This natural, ultra-high-resolution stereo vision system allows precise ranging and pose estimation in predators' brains without requiring more neural processing power. To replicate this nature-based approach, we use a convolutional neural network (CNN) to localize the receiver in image. Then, the more computationally expensive image processing steps need only be applied to the region containing the receiver.

In this paper, we contribute the following:

1. Demonstrate a novel deep learning approach to speed up stereo vision-based point cloud generation by $11.3\times$ in 4K+ image pairs.
2. Perform pose estimation with an error of $6cm$ at a target $30m$ away from the camera system.
3. Validate the system with ground experiments that accurately replicates an aerial refueling scenario.

Section II explores the background and related work. Section III outlines the design and execution of a ground experiment specifically designed to evaluate vision systems for AAR. In Section IV, we demonstrate how to implement a deep learning augmentation to the stereo vision pipeline. Finally, Section V discusses the implications of our results and how they can guide future work.

II. Background and Theory

This section is composed of four main subsections. Section II.A explains current aerial refueling techniques and limitations. Section II.B discusses existing computer vision pipelines for pose estimation. Section II.C calculates the expected benefits of using higher resolution stereo cameras. Section II.D examines how the field of deep learning has led to improvements in machine vision.

A. Aerial Refueling

Aerial refueling is the process of transferring fuel from one aircraft to another in-flight. A *tanker* aircraft transfers fuel to a *receiver* aircraft. There are two primary ways to perform aerial refueling: the boom method and the probe-and-drogue method. This research focuses on the boom method employed by the USAF. In the boom method, the receiver approaches the tanker from behind and below. The approach maneuver begins when the receiver is approximately $\frac{1}{2}km$ from the tanker. The approach ends when the receiver reaches the contact position. For this research, we approximate contact to be $30m$ from the tanker's camera system. At the contact position, a boom operator inserts the boom into a receptacle on the receiver and begins refueling. The space the receiver may occupy during

refueling is called the refueling envelope. The following constraints guide this research:

1. AAR cannot depend on precision GPS information because GPS is jammable and spoofable.
2. Receiver aircraft cannot be modified. While adding sensors and markers to the receiver would provide better results, these solutions are impractical and could adversely affect receiver performance.
3. The navigation solution must run in real time. A system that takes too long to calculate its navigation solution cannot safely control the refueling procedure.
4. To safely control the receiver, the tanker must find the exact relative position and orientation of the receiver. Inaccurate pose estimation risks putting the receiver on an incorrect flight path. This could lead to failure to connect and damage to both aircraft.
5. The refueling contact point is $30m$ from the stereo vision system.
6. The relative 3D pose estimate should have an error less than $10cm$ at the contact point.

Several efforts have been made to perform AAR. One effort to accomplish pose estimation added markers to the tanker and receiver, using geometric algorithms to calculate pose [?]. Another approach used precision GPS and an inertial measurement unit (IMU) with Kalman Filtering to estimate pose [?]. These solutions do not meet mission constraints outlined previously. Specifically, this work seeks to create a real-time, vision-only pose estimation process that has a mean 3D pose estimation error of less than $10cm$ at $30m$.

B. 6DoF Pose Estimation

One fundamental problem for computer vision systems is to derive a 3D model of the environment from 2D images. In monocular vision, information comes from a single camera or image. Zhang et. al. [?] demonstrate a deep learning process that performs object detection and pose estimation from a single camera. While their approach does run in real time, their neural network performs pose estimation on small objects that are very close to the camera. Similarly, Ferrara et. al. [?] use monocular and stereo systems to perform pose registration at ranges from $0.5m - 4.0m$. These approaches are not adequate solutions, because our problem requires a high precision solution for a very distant, large object.

Early AAR pose-estimation efforts focused on using monocular vision as a single component of a sensor-fusion relative navigation solution [? ?]. Since both of these added markers to the aircraft, they do not meet the first or second constraints outlined in Section II.A.

Stereo vision finds features in images and, after a calibration, undistortion, and rectification process, re-projects these features into 3D space relative to the cameras using epipolar geometry (for more information, see [?]). The stereo block matching algorithm locates features in both images and calculates the disparity, or distance in pixel-space, between them. Once disparities have been calculated for an image pair, the disparity map can be reprojected into space to create a 3D point cloud. Stereo block matching requires a series of pixel-wise comparisons. Increasing the number of

pixels in the image pair leads to a linear increase in computation time. Our real-time constraint imposes a limit on the resolution that can be used in for this process. Once a point cloud has been generated, there are many techniques to perform pose estimation. Since aircraft are rigid bodies, point-to-point iterative closest point (ICP) [?] was chosen for this work. However, alternate methods such as parallel ICP [?], FGR [?], or a deep learning approach [?] may provide different performance and precision trade-offs.

C. Camera Resolution and Depth Estimation

Intuitively, one expects that a higher resolution camera pair would improve depth estimation fidelity. The expected error for a stereo camera system at a given range can be calculated using

$$\epsilon_z = \frac{z^2}{bf} \epsilon_d \quad (1)$$

[?], where ϵ_z is the depth error, z is the depth, b is the baseline, f is the focal length (in pixels), and ϵ_d is the matching error in pixels (disparity values, assumed to be one). In this subsection, we simulate the error in depth re-projection for a single point using OpenCV. With properly calibrated cameras, it is possible to have a mean depth estimation error near zero at long ranges; however, the mean error is often misleading because individual depth estimations may significantly over-estimate or under-estimate an individual feature's depth. As outlined in Section II.B, the entire point cloud contributes to the accuracy of pose estimation. For this reason, we seek to decrease mean absolute error (MAE). To demonstrate the necessity for using higher-resolution camera, we used the scenario demonstrated in Figure 1, where the point being triangulated was 30m away from a stereo camera system employing a $\frac{1}{2}m$ stereo baseline.

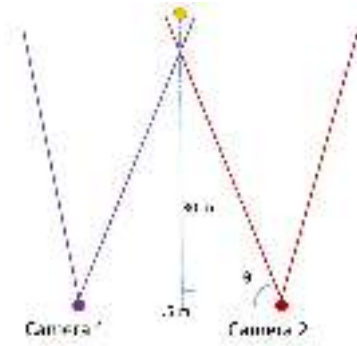


Fig. 1 scenario for single-point depth estimation

Using cameras with a fixed, 56° field of view, the cameras' resolution was varied and compared the average error in depth estimation as a function of distance from the camera baseline. With Gaussian noise and a 1-pixel standard deviation in both images, a 1280×960 image resulted in $0.4598m$ MAE in distance from the cameras. By using a higher resolution camera of 4896×3264 , a $0.38m$ MAE is achieved. This demonstrates the potential for significantly

improving the relative pose computation of a stereo vision system by increasing the resolution of the cameras. These results correspond well to the calculated error.

Table 1 Estimated depth for a feature located 30m away from stereo cameras

Resolution	Mean Depth Estimation	MAE	Calculated Error
1280×960	30.10m	1.427m	1.496m
1920×1440	30.02m	0.9435m	0.997m
3840×2880	30.00m	0.4650m	0.498m
4896×3264	30.00m	0.3844	0.3910m

D. Deep Learning in Computer Vision

Deep learning provides different trade-offs and benefits for real-time computer vision than conventional methods. CNNs require training time and labeled training datasets. However, on-line they execute in a short, constant time for each input. As [?] discusses, using deep learning with conventional techniques may provide robust solutions.

Deep learning techniques have made substantial progress in recent years towards recognizing various objects in images. Since 2010, error rates in detecting objects in an image of a large dataset has decreased from over 20% down to 1%. Architectures vary from a sequential series of convolution filters all the way to dense connections where each layer takes in the output of every previous layer as input [? ?]. Additionally, many newer CNNs have demonstrated an ability to localize the objects they identify in an image [? ? ?]. There have been experiments that show deep learning can outperform traditional methods [?]; however, conventional computer vision algorithms' intermediate representations, such as point clouds, have been shown to improve performance of deep learning solutions compared to image-only networks [? ?].

In this work, a deep CNN is trained and tested using simulated imagery. Recent literature suggests that a network capable of performing well on the simulated imagery could be trained to perform as well on imagery from physical cameras. The work in [? ?] suggests that the main issue is that the virtual camera is a different sensor from a physical camera, and domain adaptation is required when changing sensors. Goyal et. al. [?] also show that augmenting semantic segmentation image datasets with synthetic imagery can improve results. Ros et. al. [?] demonstrate the benefits of this approach. Based on these findings, we assume the CNN we trained and evaluated on virtual imagery will perform just as well when trained and evaluated on physical imagery.

III. Ground Experiment Design and Results

This section explains the experiments that were conducted to validate our previously proposed computer vision pipeline. The base pipeline comes from our previous work [?] and functions as follows:

1. Capture stereo imagery

2. Generate a disparity map
3. Convert the disparity map into a 3D point cloud
4. Use ICP to register the receiver's pose

In this section, we examine the affects of camera resolution on long range pose estimation using a ground experiment designed to mimic an aerial refueling approach. Ongoing work suggests that this experiment's residual errors at a given range closely reflect a real test flight's residual errors at the same range [?]. Section III.A describes the ground experiment we designed to mimic an aerial refueling approach. Next, Section III.B analyzes the experiment's results and their importance to future AAR work.

A. Experiment Design

The ground test compared the high-resolution cameras with lower resolution cameras mimicking an aerial refueling approach as much as possible. The experiment was run in two parts: the first with lower resolution electro-optical (EO) and infrared (IR) cameras, the second with high-resolution EO cameras.

1. Stereo Camera System

Two separate stereo vision systems comprised of two pairs of stereo EO cameras and one pair of IR cameras were employed. Using both EO and IR cameras increases the diversity of the experiments and gives additional data collection sources. The use of IR cameras provides the opportunity to validate stereo IR cameras as a viable option for stereo vision in the AAR domain.

Allied Vision Prosilica GT1290C EO cameras were chosen for the low-resolution EO stereo vision system. The GT1290Cs capture 24-bit RGB images at a resolution of 1280×960 and have adjustable focal points and apertures. The adjustable focal point has the advantage of setting the focus to infinity to maximize image clarity for objects at long distances, since a receiver in the experiments is at a contact distance of about $30m$. Additionally, the cameras do not auto focus, which interferes with the camera calibration. For the high-resolution cameras, Allied Vision Prosilica GT4905C EO cameras were chosen. These have a compatible API with the GT1290C, allowing for the same configuration except for the resolution. The cameras are capable of a full resolution of 4896×3264 ; however, to achieve $10Hz$ frame rates, the high-resolution cameras were configured to capture images at a resolution of 2448×1632 in a smaller field of view; they maintain 4K+ pixel-density if extended to the full field of view. The IR cameras chosen for the project had an image resolution of 1024×768 and the images produced are 16-bit grayscale images. Like the EO cameras, the IR cameras were also focused to infinity. All three systems have similar full fields of view and aspect ratios.

Figure 2 shows the stereo camera configuration for the low-resolution EO cameras and the IR cameras. The cameras were configured to trigger on a hardware signal controlled by the collection program. This ensures that each stereo image pair is captured at exactly the same time, and the pairs are timestamped for alignment with truth data. Images

were collected at $10Hz$.



Fig. 2 Low-resolution EO cameras and IR cameras mounted for the first experiment

2. Calibration

To perform image rectification and feature extraction, cameras must be calibrated properly. A metallic checkerboard with $30mm$ square tiles was used to capture calibration images for the low-resolution EO cameras and the IR cameras. The high-resolution EO cameras were calibrated using a larger, matte checkerboard. OpenCV's stereo calibration function was used to compute the calibration parameters as described in [?]. The checkers on the metallic board were painted using white, heat-insulating paint. This creates a temperature differential that mimics the color differential, and the same board could be used to calibrate the EO and IR cameras, however, it is easier to calibrate accurately with the larger board.

3. Truth Data

To produce accurate position truth data for the psuedo receiver that is used during experiments, a differential GPS (DGPS) system creates a relative solution between a Primary system and Secondary system. The DGPS data is used to calculate the pose truth data. Changes in orientation more than one degree would quickly force the receiver out of the refueling envelope in an aerial refueling approach, as discussed in Section II.A. Therefore, while ICP returns a 6DoF rigid body registration, the main concern is the 3D offset vector. The Primary and Secondary systems both collect and save their GPS data for post-processing. The DGPS system collects at $5Hz$. To ensure that the pose is correlated to the correct stereo image pair, the computer's clock is synced with GPS time using a Time Machine TM2000A time server, and each image is time-stamped. The two closest DGPS solutions are linearly interpolated to the time that the image pair was captured: this solution is used as the truth data for a given image pair. The DGPS system has centimeter-level error. An error this small would be sufficient for an aerial refueling connection using the tanker method. A system verified by this methodology would be directly deployable.

4. Pseudo-Tanker and Pseudo-Receiver

The pseudo tanker was designed using a wagon that could securely support the vision system, the data collection computer, the Primary DGPS system, and a power supply system. Additionally, the GPS antenna was placed above all of the equipment to mitigate multi-path or occlusion of GPS signals. Figure 3a shows the pseudo tanker with the relative coordinate frame. We use a right-handed coordinate system, defining x pointing out of the front of the cart and the cameras on the back of the cart, y as pointing to the left of the cart, and z as pointing up.

The pseudo-receiver was designed to mimic the scaled-down behavior of a generic receiver in a refueling approach. The main structure is a wing and body with patterns printed on it. Figure 3b shows a front view. Patterns are placed on the surface to mimic the paint variations, rivets, and other surface features that stereo block matching can locate on the surface of an approaching aircraft.



(a) Rear view of pseudo tanker cart with primary axes shown



(b) Front view of pseudo receiver cart

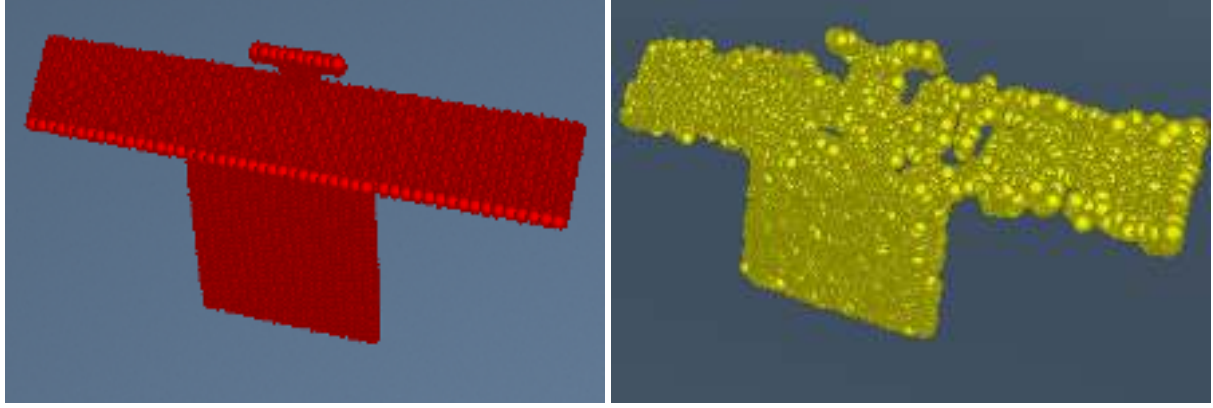
Fig. 3 Pseudo-tanker and pseudo-receiver

For pose registration, a reference point cloud (red) is matched onto a sensed point cloud (yellow) using ICP [?]. Figure 4 shows the reference point cloud for the pseudo-receiver and an example of a sensed point cloud.

5. Running the Experiment

The experiment was conducted in a parking lot to allow a large, relatively flat, open space. The pseudo-tanker remained stationary, and the pseudo-receiver was pushed towards it. Several approaches were conducted.

After the tests were conducted and truth data was obtained from post-processing the DGPS data, we applied the computer vision pipeline to estimate the pseudo-receiver's pose. Figure 5 shows an example of registration being visualized in the virtual environment. This enabled recreation of the experiment and visualization of the pose estimation in post-processing.



(a) Red Reference Point Cloud

(b) Yellow Sensed Point Cloud

Fig. 4 Reference Model and Sensed Model as Point Clouds

B. Experiment Results

This subsection discusses the results from the ground test. In Section II, we examined the expected effects of camera resolution on depth estimation. Here, we see the effects of 3D point clouds generated by different camera systems on pose estimation.

Figure 6 shows the results for one approach using the IR, low-resolution EO, and high-resolution EO cameras, respectively. In this approach, the pseudo-receiver was pushed towards the pseudo-tanker as directly as possible. Note that the IR and low-resolution EO cameras struggle to find a meaningful registration at a range of $20m$, with residual errors near $0.5m$. In contrast, the high-resolution cameras have errors smaller than $0.1m$ at ranges near $35m$.

Figure 7 shows the results for the second approach. In this approach, the pseudo-receiver was pushed a short distance and then halted for a few seconds. Since a real AAR approach might not be fully continuous, it is important to show that the technique can accurately track changes in motion as well. The results are nearly indistinguishable from the first approach, which further validates that the high-resolution cameras provide a solution accurate enough for this application.

In the approaches shown by Figure 8*, the pseudo-receiver was moved side-to-side as it approached. This was designed to imitate an approach with suboptimal conditions that required frequent correction. Each camera system performed slightly worse; however, the high-resolution system still maintained errors smaller than $0.1m$ at the target contact point of $30m$.

In all, it is clear that increasing resolution does improve pose estimation accuracy. Moreover, while the low-resolution camera system struggles to obtain meaningful registrations at $20m$, the high-resolution system can perform well at ranges near $50m$. Figure 9 shows the aggregate path estimation errors for each approach. The x , y , and z components display the mean absolute error at a given range across all three approaches. The 3D error is obtained by taking the

*The receiver briefly left the IR cameras' field of view twice in this experimental approach; this is why there is no data from $35m$ to $30m$ and an up-tick in error at $12m$.

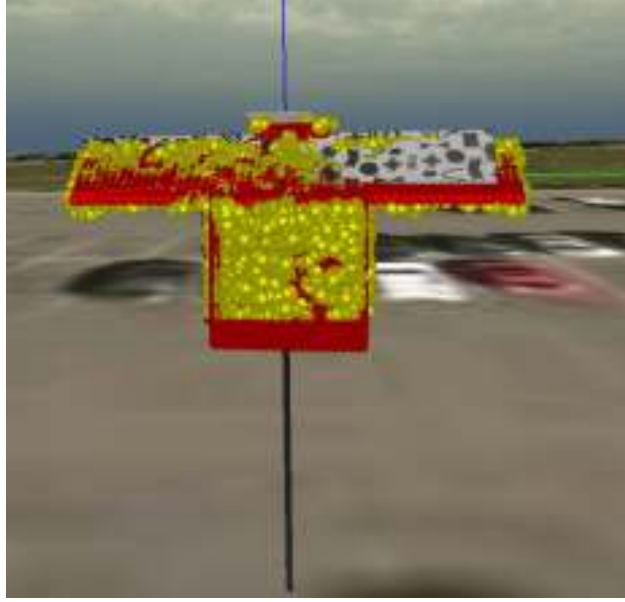


Fig. 5 Registration of the Reference Point Cloud with the Sensed Point Cloud

Euclidean distance between the sensed position and the truth position at each range. The error bars show a one standard deviation certainty associated with each mean. Importantly, we can see that the 3D error for the high-resolution cameras plus the error bound is less than the $10cm$ benchmark required for AAR.

Finally, in Figure 10 we examine the standard deviation of the errors directly. The deviation of the errors gives us important insight into the behaviors of different camera systems. While a suboptimal calibration shows residual errors for the low-resolution EO system are much higher than the IR system, examining the deviation in the errors gives the insight that the best-case scenario for those systems is actually quite similar. On the other hand, the deviation of the high-resolution EO system is much lower and much smoother as range increases.

These results indicate that our high-resolution camera system is capable of safely providing a sensed point cloud as a basis for pose estimation in AAR. However, as discussed in Section I, the increase in pixel count drastically increases the computation time required to generate a point cloud. The next section contributes a novel method to decrease point cloud generation timing with minimal impact on pose estimation fidelity.

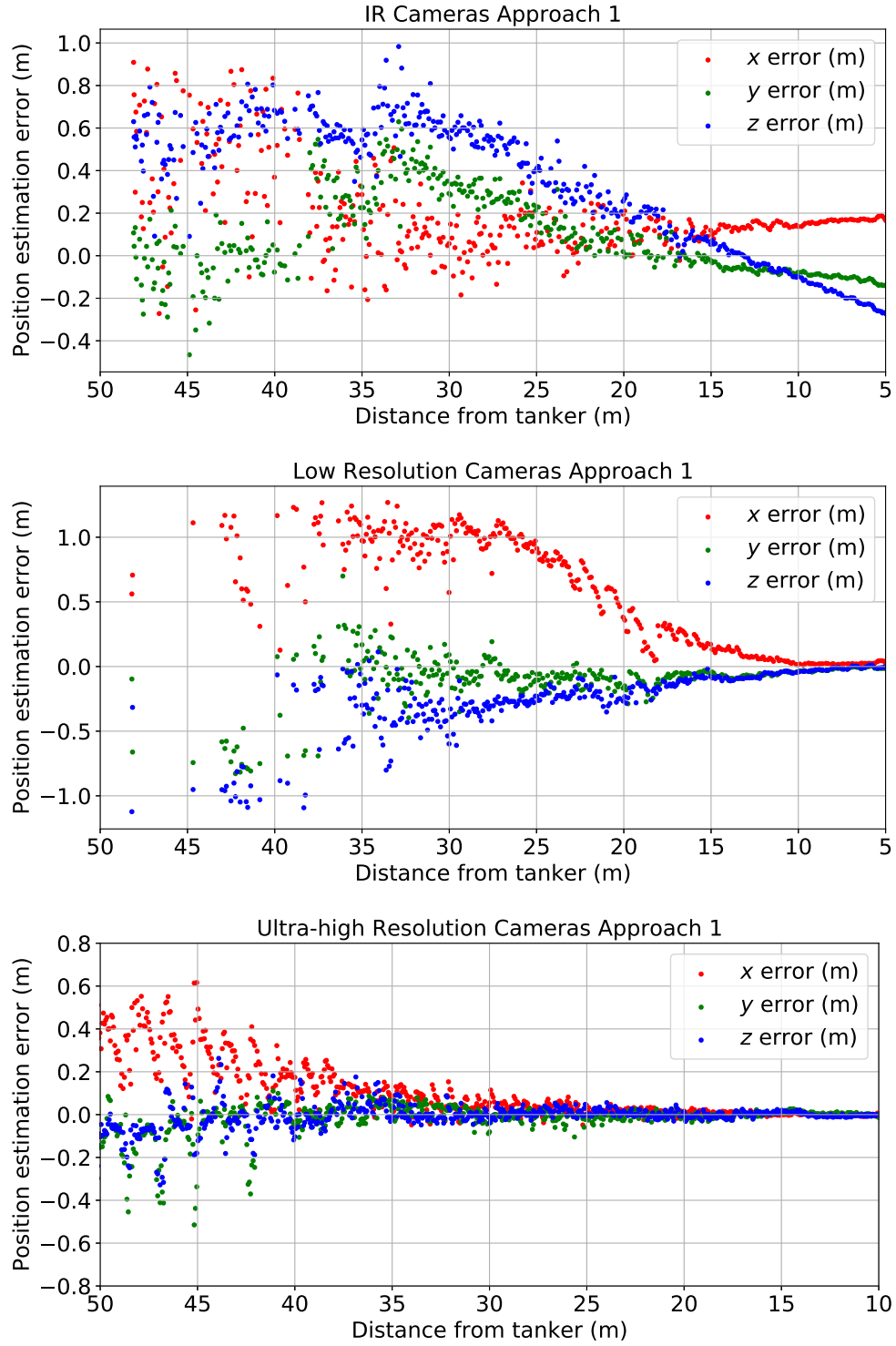


Fig. 6 Residual errors for the first approach

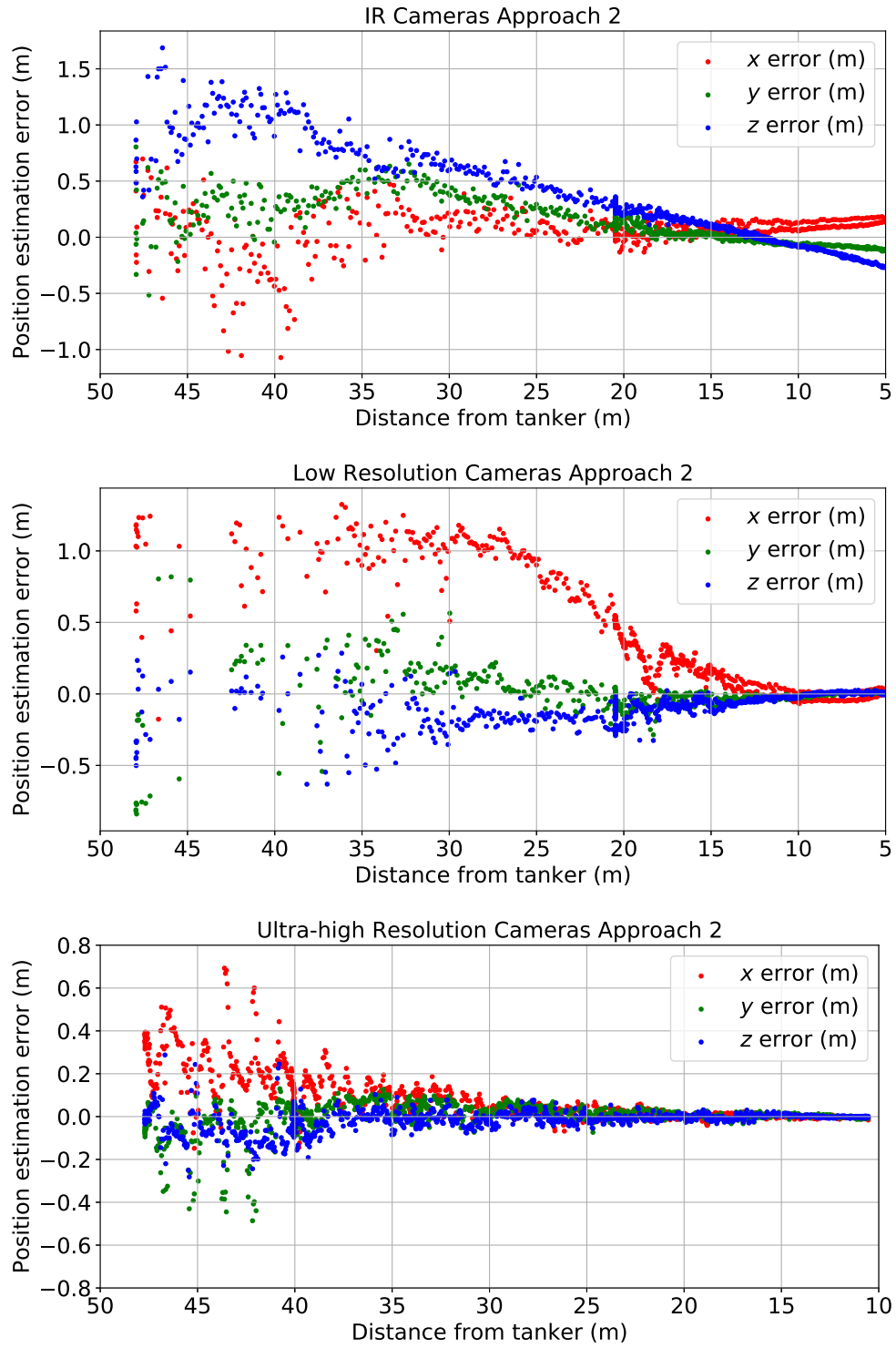


Fig. 7 Residual errors for the second approach

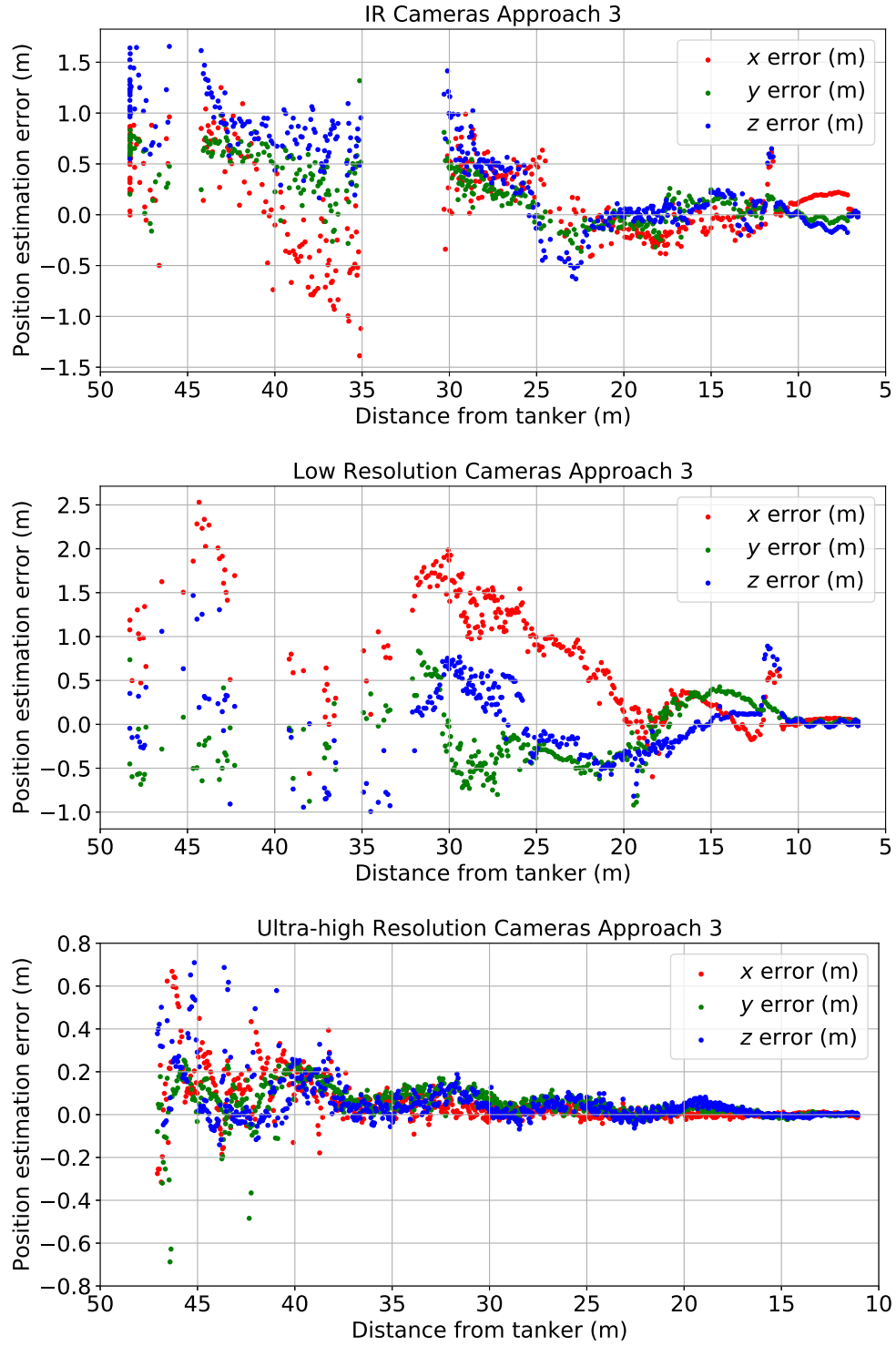


Fig. 8 Residual errors for the third approach

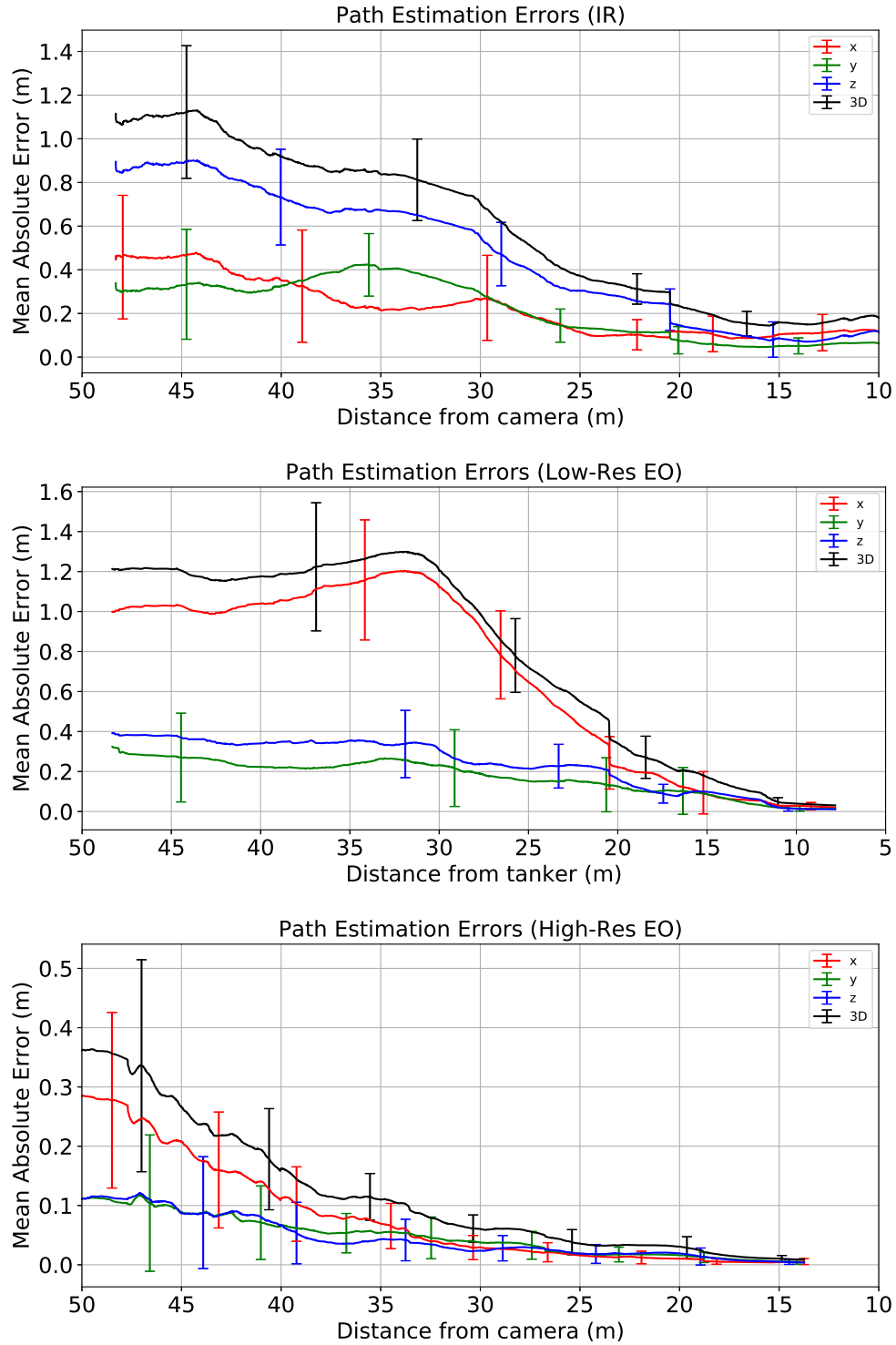


Fig. 9 Aggregate errors for each camera system across all three approaches. The 3D error is the Euclidean distance from the truth position to the sensed position.

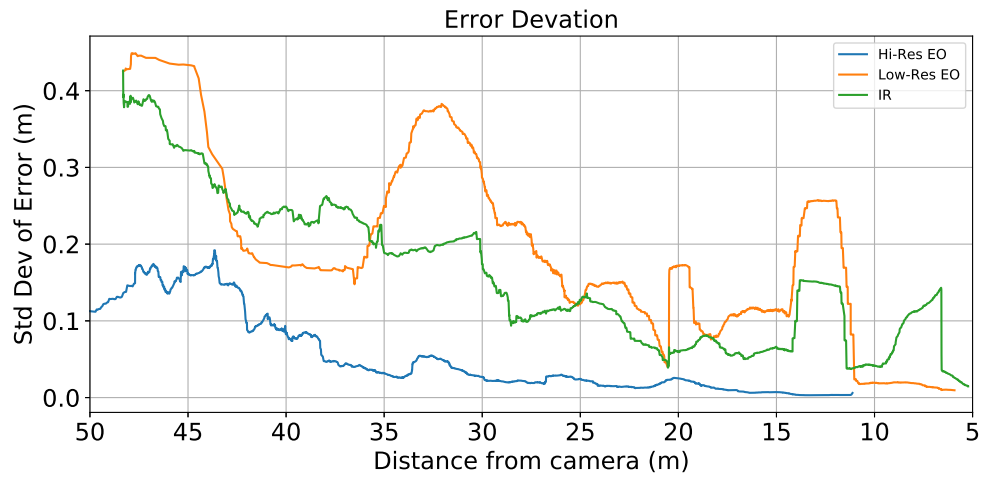


Fig. 10 Standard deviation of the error in each camera system as a function of distance. The standard deviation allows us to compare camera systems' performance to each other in best-case scenarios. For example, calibration errors lead to non-linear increases in error with distance, but these errors are consistent across multiple experiments with the same system.

IV. CNN Augmentation and Performance

To generate a 3D point cloud faster while using high-resolution imagery, we modify the base vision pipeline (shown in Section III) by cropping the high-resolution images before generating a disparity map. This leads to an improved pipeline that we now seek to evaluate:

1. Capture high-resolution stereo imagery
2. Use CNN to dynamically crop the stereo images
3. Generate a disparity map only for the region of interest
4. Convert the disparity map into a 3D point cloud
5. Use ICP to register the receiver's pose

To perform the dynamic cropping, we augment our vision pipeline with a deep learning model that is trained to segment computer-simulated imagery of a receiver aircraft. This model is deployed in a 3D virtual world refueling simulation. The model crops the stereo images to only include a tightly-bound rectangular portion of the original image containing the receiver; thus, significantly fewer pixels need to be processed using the stereo block matching algorithm. This results in a significant speedup without sacrificing precision. Section IV.A describes the 3D virtual world. Section IV.B details how our deep learning model was designed, and tested. Section IV.C explains how it was integrated into the stereo vision pipeline. Section IV.D examines the computation time required to generate a 3D point cloud, demonstrates the speedup from using a CNN to perform image segmentation, and shows that image segmentation does not adversely affect pose registration accuracy.

A. Computer Simulation

To quantify performance benefits and simulation accuracy, simulations are performed in the AFTR Burner Engine. The AFTR Burner Engine is a general purpose 3D graphics engine based on OpenGL. In this work, we use geometrically accurate models, high quality textures, and realistic lighting to replicate real refueling approaches and generate synthetic imagery. This is the same simulation environment that several researchers have used [10, 11] for their AAR experiments. The cameras in the simulation have the same resolution and field of view as their physical counterparts used in the ground experiment. To remove jagged edges caused by OpenGL's rendering pipeline, we perform multi-sample anti-aliasing (MSAA). Calibration values are calculated from known simulation parameters (resolution, field of view, and relative offset between the cameras) as described by Kaehler and Bradski [12]. To verify that increased resolution improves pose estimation, a simulated refueling approach was conducted in the virtual world using cameras at different resolutions and the fidelity of the pose estimation is compared.

B. CNN Design

For this research, a basic deep CNN architecture was created with the goal of placing a rectangular bounding box around a receiver's location in an image. There are many existing architectures, such as YOLO [?], that could perform a similar function; however, most image segmentation networks are designed for more general purposes. This one is designed for high-fidelity, real-time segmentation on a specific target. We then demonstrate how this deep learning augmentation improves our computer vision pipeline, yielding large performance benefits. The remainder of this subsection explains how the network was trained and evaluated. Its image segmentation performance is then discussed.

1. Data

The dataset for this project was created using the AFTR Burner engine described in Section IV.A. In the simulation, a virtual receiver was placed at random, uniformly distributed locations within the camera's field of view at distances between $20m$ and $100m$ and its orientation was randomly adjusted by small amounts to ensure diversity. One of several background images of real landscapes from aerial views was placed in the background. The engine used a virtual camera to capture a 1280×960 resolution image of the simulated scene. Next, the simulation was modified to re-skin the receiver in a flat, distinct color. After a pair of images was captured, the background was changed and the receiver was moved. Figure 11 shows an example of an image pair. For this research, 5,500 input/truth image pairs were generated. The project used 5,000 pairs for training and validation and 500 pairs for testing. The test set was generated using different background images that were unseen in training.

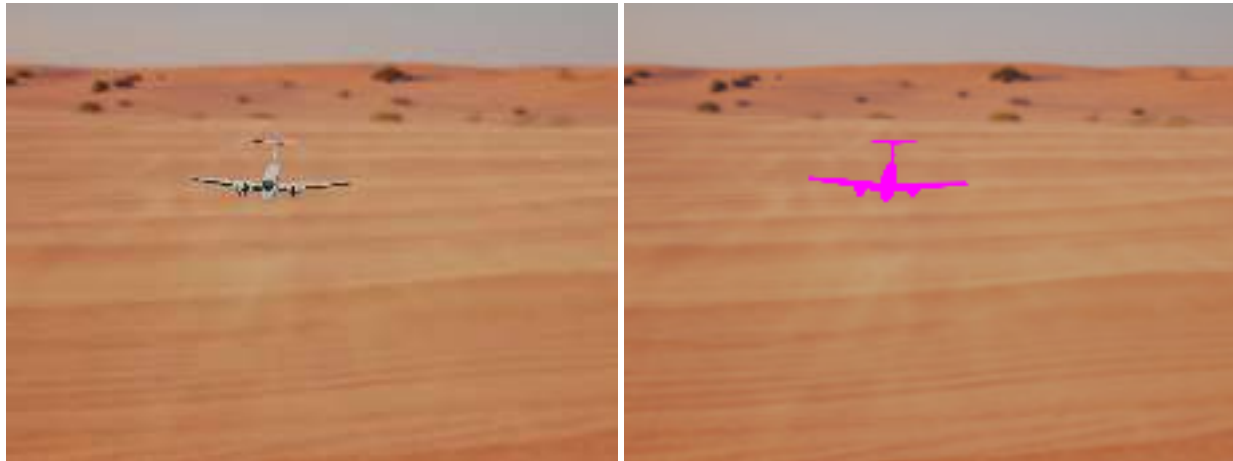


Fig. 11 Example training image pair captured from AFTR Burner

To create the truth data, a mask was applied to the image to locate the pink skin (shown in the right picture of Figure 11) on the receiver in the truth images. The minimum and maximum pixel coordinates were used to calculate the center x , center y , width, and height of the box in pixel space. These were saved in a CSV with the associated image number. Each of the training images was blurred using a 3×3 low-pass blurring filter to help prevent the model from

over-fitting potential sharp edges in the simulated imagery.

The model was further designed to perform image segmentation specifically for AAR. High-resolution cameras can often capture at higher frame rates in grayscale than in color; however, full resolution is not necessary to localize the receiver. Down-sampling images allows for accurate localization with smaller networks. For these reasons, training images were down-sampled to 512×386 and converted to grayscale. Brightness was varied in each image between 5% and 300% to help feature selection become less dependent on specific lighting. Pixel values were then rescaled to floats between 0 and 1. For testing, the pixel values are rescaled but not otherwise augmented.

2. Model

This research used a new deep CNN model. The model has 16 convolution layers and two fully connected layers before output. It takes an image as described in Section IV.B.1 and outputs regression values for the bounding box as center- x , center- y , width, and height values, normalized to be in range $[0, 1]$ as a proportion of the original image.

Batch normalization is performed at each layer. The leaky ReLU function[†] serves as the activation function for each layer. There is a 40% dropout before each fully connected layer and a 20% dropout before the output layer. These regularization techniques help prevent over-fitting the training data. The training loss function is the MSE for each predicted value (x , y , width, height, scaled $[0, 1]$).

The workstation training and evaluating the model had an Intel i7-7820X processor, 96GB of main memory, and an Nvidia 1080Ti GPU. The model trained in less than three hours. By functioning on a consumer-level computer, the model demonstrates that it can be used in practical settings.

3. Testing

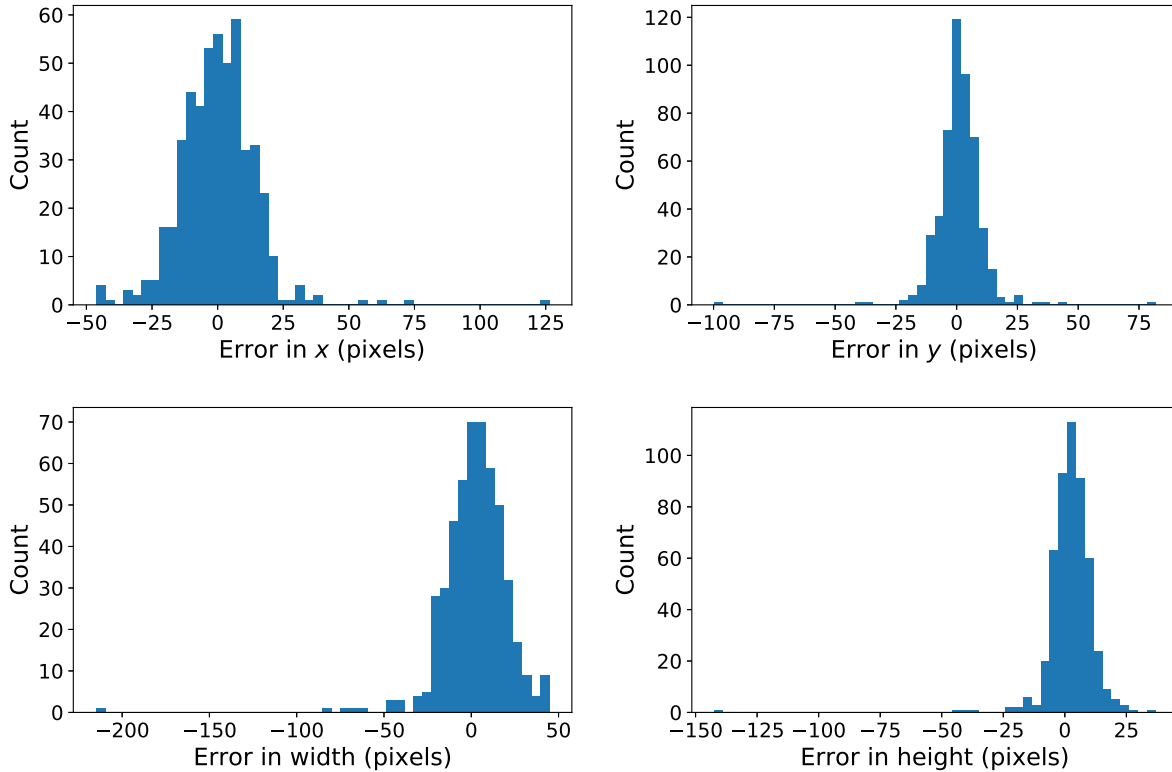
Test results are calculated using a set of 500 images that were generated and labeled as described in Section IV.B.1. The backgrounds used in these images are totally different from the backgrounds used in training, and are not seen by the model prior to the testing. The model's prediction can be directly compared to the truth data. To quantify the model's performance, we measure the distance between the predicted bounding box's center and the true bounding box's center in pixel space. We measure the CNN's error in pixel space because stereo block matching generates disparity values in pixel space as well. One of the best ways to assess accuracy for this application, though, is by viewing the images directly. Section IV.B.4 shows examples of bounding boxes generated by the model and compares these to the truth data of the original images.

Table 2 Errors for the deep learning model (in pixels, images at 1280×960) on the test set

	x	y	$width$	$height$
RMSE	14.72	9.99	19.41	10.43
MAE	10.46	6.21	13.18	6.31

4. Network Performance

Table 2 shows the quantitative measures of the network’s performance. We evaluate RMSE and MAE of the predicted bounding box compared to the truth bounding box in pixels. A few outliers slightly skew the errors; however, most are very near zero. Figure 12 shows the distribution of errors in x , y , width, and height. On average the network performs very well. Figure 13 shows the amount of the true bounding box that the predicted bounding box covers as a function of distance between the camera and the receiver. In the entire test set of 500 images, there is only one image where the predicted bounding box does not overlap the truth bounding box. Moreover, on average the model’s prediction overlaps 90% of the true bounding box. These results demonstrate stable behavior throughout the refueling approach.

**Fig. 12** CNN predicted bounding box errors in x , y , w , h (in pixels)

Section IV.B.4 shows four examples of the CNN’s typical performance. These are consistent with the model’s

†

$$f(x) = \begin{cases} 0.1x & x \leq 0 \\ x & x > 0 \end{cases}$$

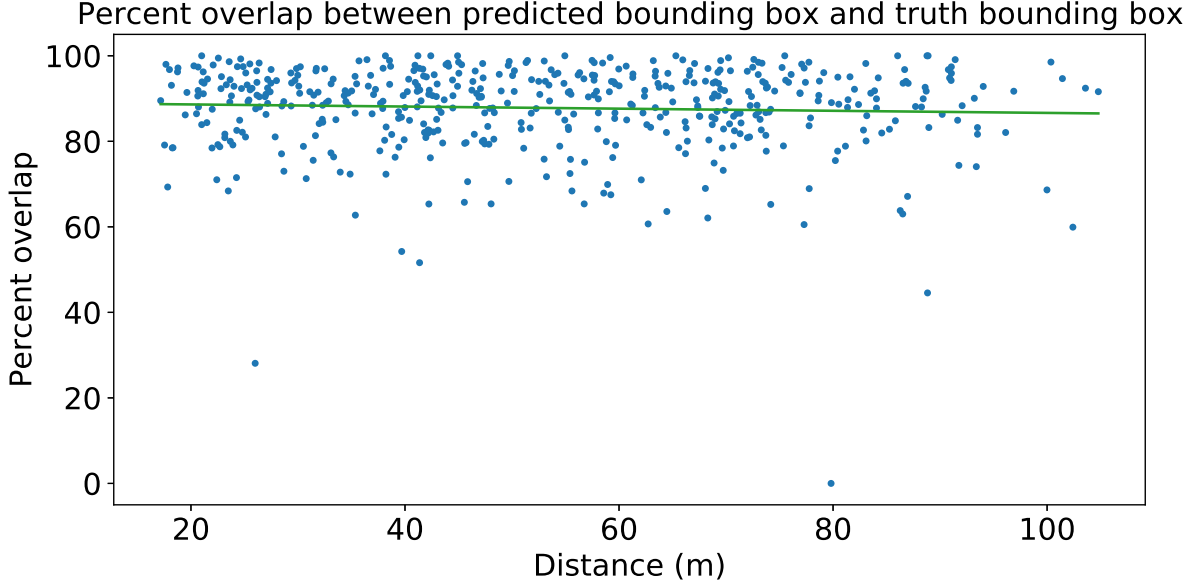


Fig. 13 This measures the percent of the true bounding box that the predicted bounding box covers and shows a best fit line for convenience

performance across the test data set. The model can also be used without transfer learning or other modification to evaluate images from a real flight test. Figure 15 shows an example of the model directly being used on imagery from a physical camera. Unfortunately, we do not have a large enough labeled dataset using physical imagery with varied views of a receiver to compute full evaluation metrics.

C. CNN Application Procedure

To provide a speedup to AAR pose estimation, the CNN’s bounding box is used to reduce the computational cost of stereo block matching. Once the stereo images are captured, the left image is down-sampled from the original resolution to 512×386 and passed as input to the CNN. While it would be possible to perform bounding on both images, the disparities at $30m$ are only a few pixels. This means that the error from assuming both bounding boxes are the same is small enough that it does not appreciably decrease the CNN’s performance. Additionally, this means the network only runs once, saving valuable computation time. The bounding box is then used to mask a pre-computed rectification map (Kaehler and Bradski give an in-depth explanation for the rectification process [?]). The captured images are remapped using this now-cropped rectification map into a final pair of rectified, undistorted, and cropped images. These images are then passed into OpenCV’s stereo block matcher to generate a disparity map. Finally, the disparity map is reprojected into 3D space for use as a point cloud for pose registration. To compare the previous pipeline with the new one, we collect data on the precision of the pose estimation process for a simulated approach and also time the point-cloud generation process for stereo camera pairs at a variety of resolutions.

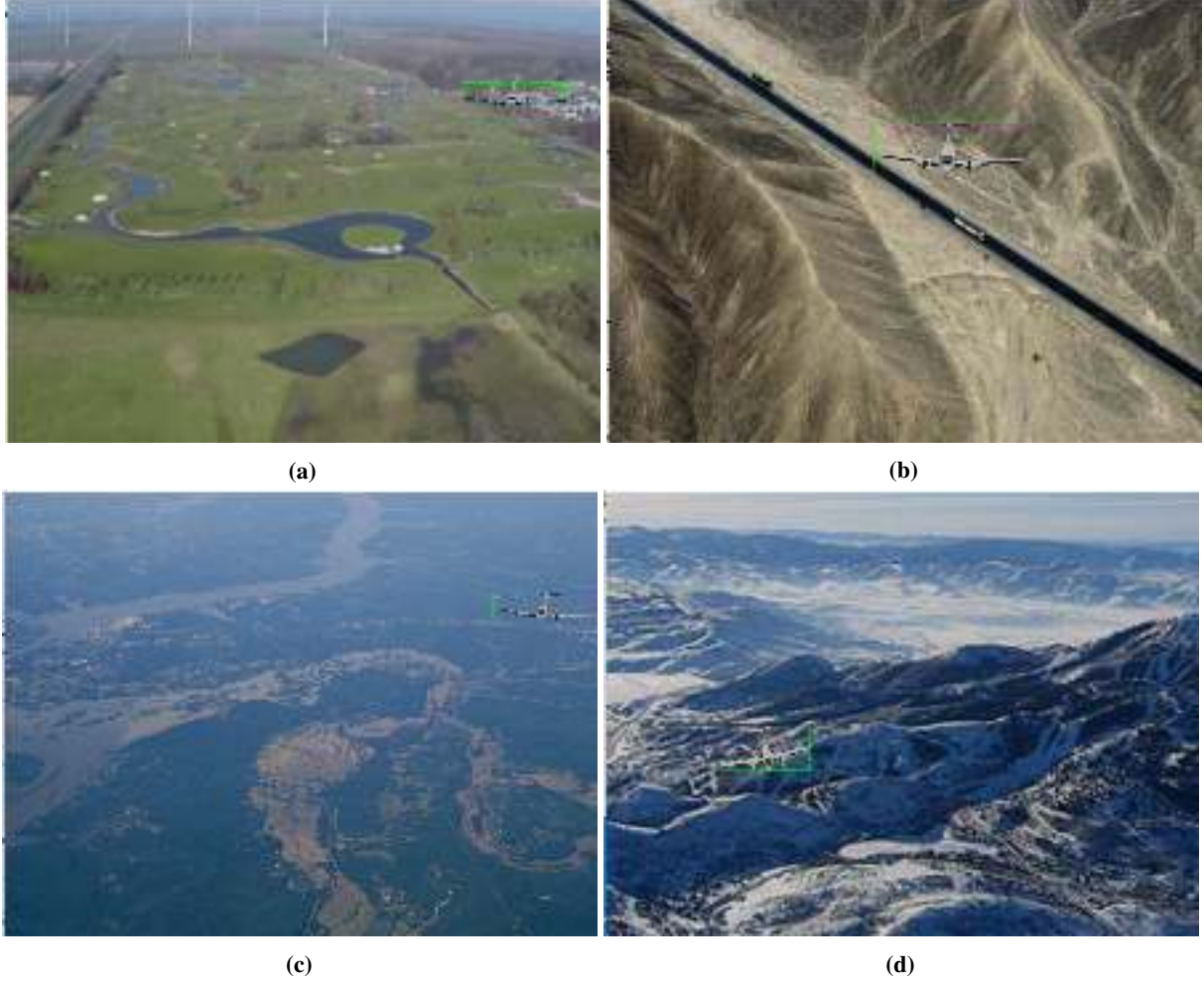


Fig. 14 Examples of CNN model performance. The green box represents the ground-truth bounding box. The purple represents the network’s predicted bounding box.

D. Point Cloud Generation Timing and Pose Estimation Precision

Table 3 Point-cloud generation time with and without using the CNN

Resolution	CNN Off	CNN On	Speedup
1280×960	36.13ms	10.57ms	3.42
1920×1440	76.77ms	14.67ms	5.23
3840×2880	357.49ms	35.03ms	10.19
4896×3264	524.28ms	46.52ms	11.27

This subsection evaluates the CNN’s performance at improving the pose estimation process for AAR. First, we demonstrate that the CNN-augmented pipeline requires much less time to generate a 3D point cloud. Next, we examine



Fig. 15 Performance example for the CNN model with real flight test imagery.

the effects of using the CNN-augmented pipeline on the precision and deviation in pose estimation accuracy. Table 3 shows the time required to generate a 3D point cloud at four sample resolutions with and without the CNN augmentation. It is clear that the CNN provides a substantial speed-up at all resolutions. Creating a 3D point cloud for the 1280×960 image pairs is $3.6\times$ faster with the CNN augmentation. When the resolution increases to 4896×3264 , the CNN gives an $11.3\times$ speedup.

Finally, it is important to verify that the CNN augmentation does not adversely affect pose estimation. As with the ground experiment, Figure 16 shows the 3D path estimation error over the approach. Figure 17 shows the standard deviation of the 3D error for the 1280×960 camera system and the 4896×3264 camera system with and without CNN augmentation to the vision pipeline.

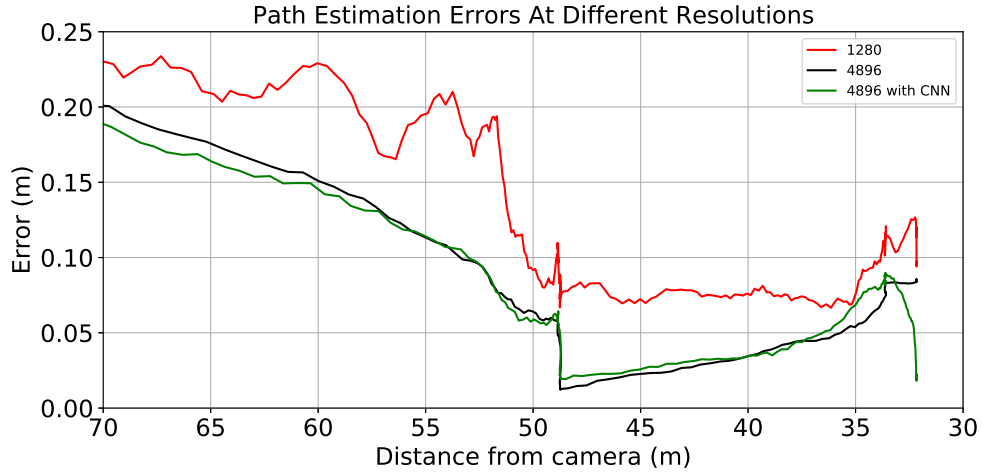


Fig. 16 Path estimation error (Euclidean distance from truth position to sensed position in meters). Note that the low-resolution system's solution, even in the simulation, is consistently worse than the high-resolution system. Also note there is not an appreciable difference between the high-resolution systems error with or without the CNN augmentation. This shows the system can reliably perform as well as the original pipeline, while also gaining a large speedup.

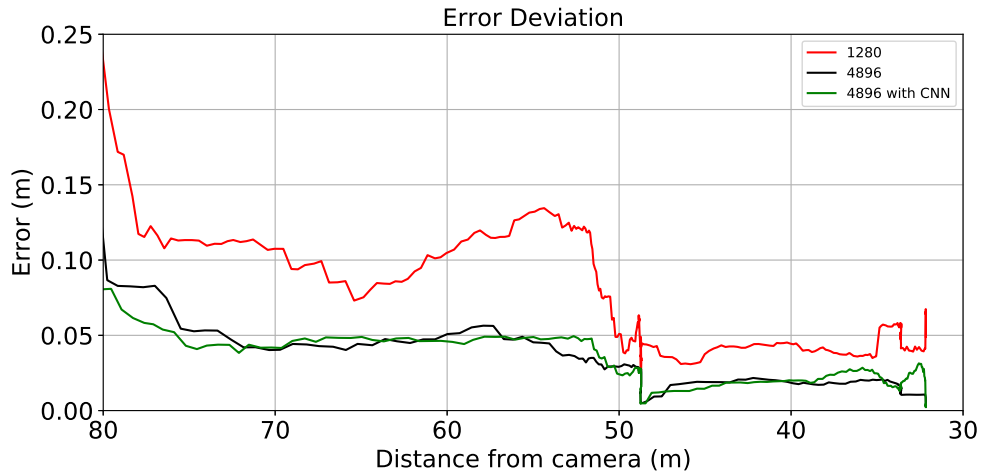


Fig. 17 Standard deviation of the error of each camera system's pose estimation as a function of distance. There is no appreciable difference between the high-resolution system with and without the CNN augmentation, further validating its viability.

V. Conclusion and Future Work

In this paper, we validate our stereo vision pipeline to perform pose estimation for AAR with a novel ground experiment. The system consistently achieves 3D pose estimation errors of less than *6cm*. Based on these results, a stereo camera system with adequate resolution can safely control a receiver in the refueling envelope to make and maintain contact. However, high-resolution imagery comes with a computation-time cost.

Next, we outline a computer vision pipeline that combines conventional stereo vision with deep learning to greatly accelerate the process of generating a 3D point cloud of the receiver. We further verify that the speedup does not decrease the precision gained from using high resolution stereo imagery. While this system was developed specifically for AAR, any real-time computer vision application could benefit from its use. For example, a CNN can identify and label several objects of interest in a stereo image pair and then perform the pose estimation process quickly on each of them. Since the point clouds are generated from finely cropped images, the resulting 3D point clouds are already semantically segmented. This technique could be used provide benefits for many computer vision applications, including autonomous vehicles, robot navigation, or structure from motion.

The greatest remaining limitation for pose estimation speed is point cloud registration. Using a parallel ICP or development of a faster point cloud registration algorithm will be important for any future efforts to increase pose estimation rates. Further study could be done to determine if pixel-wise image segmentation could yield even greater speedups for the AAR domain. A pixel-wise mask could be used to crop the images similarly to the rectangle our method uses. Additionally, error in pose estimation and image segmentation could likely be greatly reduced by leveraging the time-series nature of most visual relative navigation tasks. For segmentation, leveraging a recurrent neural network (RNN) to perform image segmentation may allow better tracking of objects. Future work to improve pose estimation may employ a Kalman filter.

Funding Sources

This work was funded by the Air Force Research Laboratory Aerospace Systems (AFRL/RQ).

Disclaimer

The views expressed in this document are those of the authors and do not reflect the official policy or position of the United States Air Force, the United States Department of Defense or the United States Government.

Acknowledgments

Special thanks to Capt Michael Crawl for his help in setup and data collection for the ground experiment.