Aircraft Wing Shape Analysis by On-board Cameras and Deep Learning

A Synthetic Proof of Concept

Alexander Portiankin[1], Ido Plat[1], Ido Imanuel[1], Aviad Zabatani[3],
Prof. Ron Kimmel[1], and Prof. Daniella E. Raveh[2]

[1]Department of Computer Science, The Technion
[2]Department of Aerospace Engineering, The Technion
[3]Intel, RealSense Division, Israel Design Center, Haifa.

May 17, 2021

Contents

3 Chapter 1

Introduction

4 Chapter 2

Background

2.1 Parametric Representation of a Deformed Shape 4

2.2 Overview of the Experimental Wing 6

2.3 Overview of Synthetic Wing Representations 7

8 Chapter 3

Method

3.1 Synthetic Simulation Targets 8

3.2 Metrics 9

3.3 Dataset Generation 9
3.3.1 Textures 10

3.4 Validation Split 11

3.5 Noise 12

3.6 DNN Overview 13
3.6.1 Numerical Stability 13
3.6.2 Baseline 13

14 Chapter 4

Analysis

4.1 Theoretical Frequency Range 14

4.2 Texture Optimization 15

4.3 Camera Optimization 18
4.3.1 Camera coordinates optimization 18

4.3.2 Channel Information Optimization 20
4.3.3 Camera count optimization 22

4.4 Theoretical Regression Error Under Lab Conditions 26

28 Chapter 5

Conclusions and Further Research

5.1 Conclusions 28

5.2 Further Research 28

29 Chapter 6

Bibliography

31 Chapter A

CAD Model approximation using the FE Model

33 Chapter B

MRS Analysis - A Note For The Experimental Phase

Chapter 1

Introduction

The main drivers in today's aircraft design are performance improvements, reduction of fuel consumption and harmful emission. These goals can be achieved in a straightforward fashion with lightweight, large wingspan designs. However, such configurations are inherently more flexible and susceptible to adverse aeroelastic phenomena including reduced control authority, increased maneuver loads, excessive response to atmospheric turbulence, and flutter instability. The immediate remedy for aeroelastic problems is stiffening the structure. This, however, comes at the cost of additional structural weight and, ultimately, a penalty to the systems performance. Over the years, and with advances in aircraft control methodologies, studies have shown that a lightweight, flexible air vehicle can be controlled to mitigate flutter instability [1], alleviate gust response [2], or achieve optimal performance while minimizing adverse aeroelastic effects [3–6].

Most of the commonly used means to control aeroelastic phenomena rely on manipulating the aerodynamic forces achieved by static or dynamic activation of control surfaces. The control surfaces effectively modify the flexible wings' deformed shape, such that the resulting flow regime and pressure distribution generate favorable aerodynamic forces. However, as of yet, direct measurement of the deformed shape of a wing is not available. Therefore, active control systems provide feedback based on local measurement of acceleration (e.g. wingtip acceleration), or strain (e.g., a strain-gauge at the wing root), but not directly on displacements. Some numerical studies on aeroelastic control assume that the deformed shape is known. It could be in the form of wingtip displacement [5], or modal deformation [6, 7]. However, neither is readily measurable. It is noted that local deformations can be obtained from accelerometer measurements by temporal integration. Still, this information is local and limited to a few points, and the double temporal integration of accelerations to obtain deformations introduces errors.

Recent studies suggested that a detailed deformed shape of a structure could be obtained from *strain-data* measured in optical fibers [8–10]. Fiber-optic sensing is used in civil engineering, aerospace, marine, and oil and gas, and their inherent capabilities make them highly suitable for embedding in aerospace systems and specifically in wings. While Fiber-optic sensors (FOS) appear to offer accurate information on wing deformations, their usage is not suitable for small air vehicles. The smallest FOS interrogator weighs approximately 1.5 kg, making for a bulky, heavy, and expensive system to fly.

The primary goal of this research is to explore whether we can extract accurate information on the deformations of a flexible wing in flight using inexpensive, lightweight, off-the-shelf cameras.

The study proposes developing a methodology based on Deep Learning, which receives as input images of a deformed wing, as captured by wing-mounted cameras, and produces, as an output, a set of parameters defining the deformed wing shape. The research targets are to test the methodology numerically and experimentally using a wind-tunnel model of a flexible wing that has dynamical properties typical to those of realistic aircraft wings.

If proven viable, such camera systems can be carried on-board lightweight vehicles, providing vital information for controlling and mitigating aeroelastic phenomena. This will lead to ever-lighter configurations, better performance, and lower environmental impact, which rely on active-control based on direct deformation measurement. In this study, we make an important first step - we implement and test the proposed methodology using *synthetic, computational data*. We simulate a large dataset of 2D wing images, along with their corresponding deformation parameters and train a neural network, validating it on many unseen examples. We show very promising results in preparation for the experimental phase where this methodology will be empirically tested under lab conditions.

Chapter 2

Background

2.1 Parametric Representation of a Deformed Shape

The deformed shape of a structure can be compactly represented as a combination of geometric *shapes* in 3D according to

$$\{u\} = \sum_{i=1}^{N} \Phi_i \xi_i \tag{2.1}$$

where $\{u\}$ is the vector of displacements over the structure, Φ_i is the i-th shape, and ξ_i is its corresponding coefficient. An optimal choice of shapes are the structure's *natural mode shapes*, as they satisfy the governing differential equation that determine the deformed shape, as well as all boundary conditions. An example of such modal shapes is in figure 2.1.

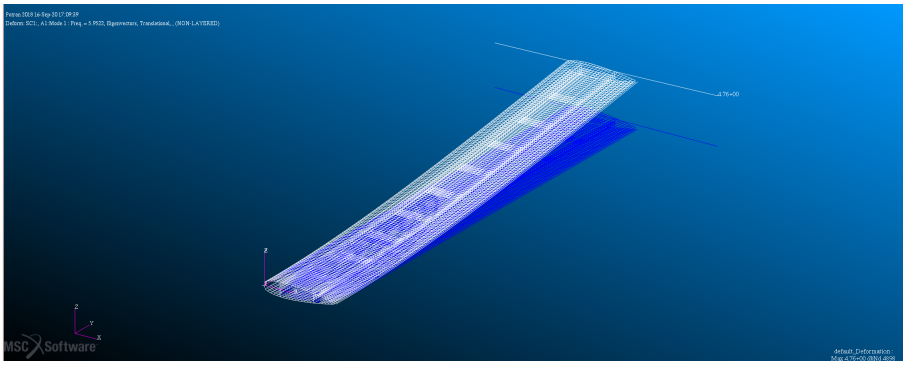
The natural mode shapes are a property of the elastic structure and can be computed from a structure's finite-element (FE) model by solving an eigenvalue problem [11]. These modal shapes are validated and fine-tuned experimentally using Ground Vibration Test (GVT) procedures.

Expressing the deformed shape of an elastic structure, e.g., an aircraft wing, in terms of mode-shapes (Eq. 2.1) is referred to as *the modal approach*, and the modal participation coefficients, $\{\xi\}$, are the *modal displacements*. A structure has an infinite number of degrees-of-freedom (DOFs) and correspondingly, natural modes. A FE discretization of a realistic structure, e.g., a wing, typically has tens of thousand DOFs and natural modes. However, in the case of wing deformation, where we are typically interested in the large, global deformations rather than in the small, local ones, the deformed shape can be represented with only a few, low-frequency, modes. This is a great advantage as the use of the *modal approach* substantially reduces the number of DOFs of the problem in several orders of magnitude.

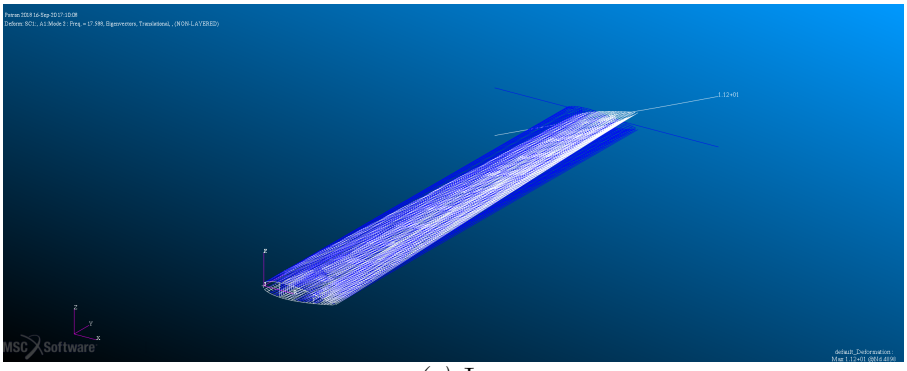
In the current study, finding the deformed shape of a wing refers to the process of identifying the modal displacements, $\{\xi\}$, associated with a given set of mode shapes, $\{\Phi\}$, based on 2D images of the deformed wing. We propose to do this by training a deep neural network (DNN) to associate between 2D images and modal displacements. Given a dataset of images, in which an image $\boldsymbol{X}_i$ is associated with a known modal displacement vector $\boldsymbol{\xi}_i$, the trained DNN will identify $\boldsymbol{\xi}_j$ of a newly introduced image $\boldsymbol{X}_j$.

Figure 2.1: FE representations of the first four natural modal shapes of the wing; In blue - undeformed model; In gray - mode shape

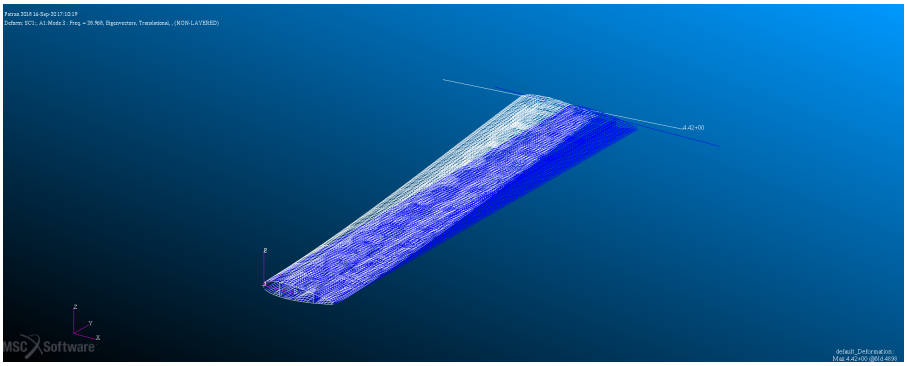
(a) Φ_1



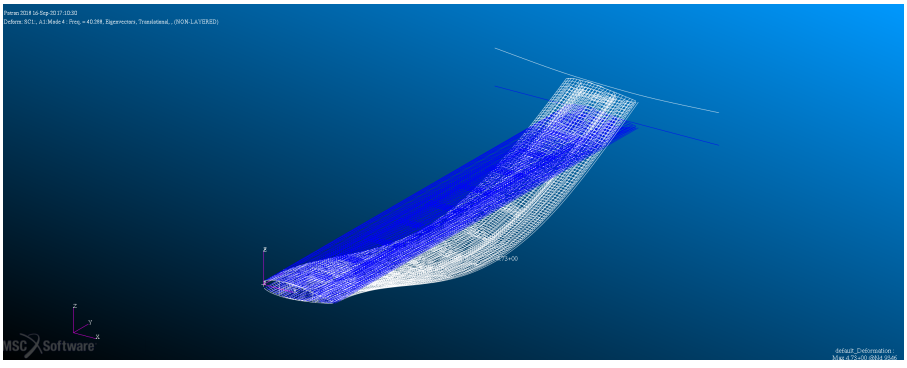
(b) Φ_2



(c) Φ_3



(d) Φ_4



2.2 Overview of the Experimental Wing

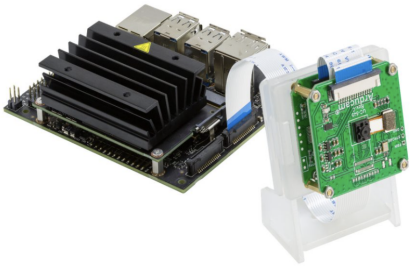
In this study, we test the methodology numerically in preparation for the sequel where we will test it experimentally using a wind-tunnel model of a flexible wing similar to the one shown in figure 2.2.

Figure 2.2: A flexible wing model in the wind-tunnel. Wing deformations tracked by optical IR markers

This wing was designed in the Aeroelasticity Lab in the Technion [10]. It serves as a small and easily manufactured model wing, able to fit in the Technion's wind tunnel while emulating the structural properties, and behaviour of larger, realistic wings. Its structure is modeled in FE software, and analyzed for its deformed shape under various aerodynamic loading conditions. Computational data include the wing's natural mode shapes and frequencies, and deformations, in physical and modal coordinates, under various loads. The computational data comprises of a large set of deformation cases to serve the learning process. In the upcoming experimental study discussed, the wing will be manufactured and tested in a wind-tunnel experiment, where it will be deformed due to the aerodynamic loads at different air-speeds and angles of attack. In the wind-tunnel test, 2D images of the deformed wing will be taken by one or more small, cheap, global-shutter cameras, similar to the Arducam in figure 2.3. Reference wing deformations will be captured by an OptiTrack[1] motion recovery system (MRS) comprised of three cameras tracking optical infrared (IR) markers glued to the wing surfaces. Figure 2.2 shows two of the three cameras circled in red. Experimental data will include the wing's natural modal shapes, the dominant frequencies, and the spatial locations of the IR markers on the wing in various flow conditions. The modal displacements can be estimated using this data. A typical sparse distribution of the IR markers over the wing is seen in figure 2.2 (the wing-tip center marker is circled in grey).

[1] http://optitrack.com

Figure 2.3: Arducam OV7251 camera module; 40mm x 40mm, 8 gr

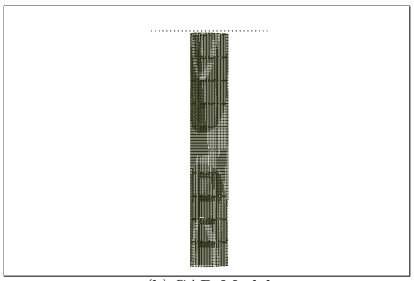


2.3 Overview of Synthetic Wing Representations

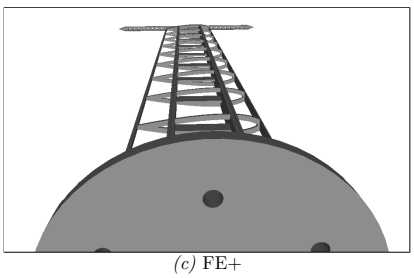
In the study of modal analysis, two digital representations of the wing are necessary, a Final Element (FE) representation to simulate its physical behaviour in software and a detailed Computer Aided Design (CAD) that is 3D-printable. The FE model is visually dissimilar to the CAD model, doesn't include a visual tip and has no face information and thus can't be directly used for generating $\{\boldsymbol{X}_i\}$. The CAD model has no material properties and thus can't be used to simulate the wing's behaviour. Thus, to generate the synthetic dataset we create a modified FE model (FE+) to visually approximate the CAD model using the FE model. These models are presented in figure 2.4 and the specifics of this modification are available in Appendix A.

Figure 2.4: The FE model, the CAD model and the approximation of the CAD model using the FE model (FE+)

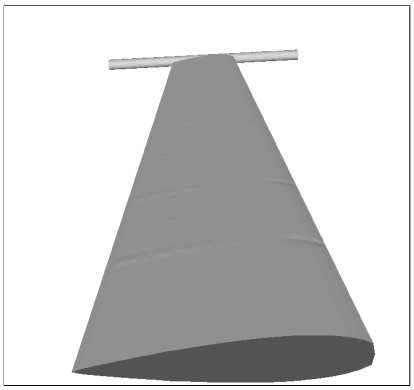
(a) FE Model



(b) CAD Model



(c) FE+



Chapter 3

Method

The goal for the current study is to identify the modal displacements relating to a deformed wing shape from 2D pictures of the wing synthetically created to simulate a small on-board camera. The functional relationship between 2D pictures of the deformed wing and the modal displacements will be established via a trained DNN, where the training data will include a set of cases of deformed wing shapes projected onto a 2D plane to create the needed images, and their related modal displacements as ground truth labels.

3.1 Synthetic Simulation Targets

We turn to discuss in detail the research questions addressed by this synthetic computational synthesis. We enumerate them as follows:

1. *Theoretical Frequency Range* - The nature of aerodynamic loading is such that it excites the *global* wing motions, which are described by the low-frequency modes (e.g. first-bending, first-torsion, etc.), more than it excites *local* displacements, which are described by the higher, local, modes. Additionally, the intrinsic structural damping of the wing contributes to faster decay of the high-frequency modes. As a result, a wing deformation can be accurately described (for aeroelastic control purposes) only by its first few low-frequency modes. Furthermore, a typical infrared (IR)-sensor based motion recovery system (MRS), such as the OptiTrack system used in the Aeroelasticity Laboratory, introduces substantial noise due to the measured locations of the IR-sensors. This, together with a limited sampling rate in the order of a few hundred Hz, heavily degenerates higher frequency measurements. We recognize that a fully computational measurement does not suffer from the same failings, and as such, we turn to question of *what is the maximal theoretical spectral range recoverable from 2D projections?* We address this by measuring signal to noise statistics for each recovered mode, empirically quantifying the theoretical confidence of retrieval score for each discrete frequency. This is discussed in section 4.1.

2. *Texture Optimization* - One central decision to be made is of which texture is to be fitted to the wing upon a realistic measurement. Preferably, we require a texture that encodes as much geometric information as possible, so heights and distances may be readily decoded from the image. By utilizing a variety of synthetic textures, we plan to empirically estimate the contribution of such textures to a realistic inference setting in the sequel. This is discussed in section 4.2.

3. *Camera Optimization* - In preparation for a realistic placement, we must determine the number of cameras placed, their optimal location and orientation along with the preferred channel information. We note, for example, that information inherent in one view may vastly differ from another. Likewise, a noisy depth channel may be insubstantial when compared to pinpoint RGB information. Aerodynamic and photometric considerations introduce additional restrictions, resulting in a constrained optimization problem that must be solved in order to maximize inference accuracy.

   This is discussed in sections 4.3.

4. *Theoretical Regression Error* - Estimate the minimal theoretical error of a modal displacement regression from 2D views under laboratory conditions. Given some ground truth displacement $\boldsymbol{\xi}_i$, and our estimate $\hat{\boldsymbol{\xi}}_i$, we measure their discrepancy via multiple suitable error metrics $\left\{ d\left(\boldsymbol{\xi}_i, \hat{\boldsymbol{\xi}}_i\right) \right\}$. This serves to supply a rough empirical estimate of the amount of information intrinsic to only partial and local 2D projections of the wing. This is discussed in section 4.4.

3.2 Metrics

In order to achieve targets 2, 3, 4 from section 3.1 we must define suitable error metrics. No significant research was done into Deep Learning based methodologies for modal analysis, hence a common evaluation metric for this sort of task does not exist. We propose, and track, three different error metrics between $\boldsymbol{\xi}_i$ and $\hat{\boldsymbol{\xi}}_i$.

1. *100% metric.*
   The L2 error between the vertices of the point clouds created from $\boldsymbol{\xi}_i$ and $\hat{\boldsymbol{\xi}}_i$, normalized by the vertex count. This metric is the obvious choice, it attempts to relate to the error benchmark used by the MRS. This metric will be referred to as the *100% metric*. This metric presents two issues.

   - A large part of the wing is nearly stationary throughout all entries, thus reducing the useful information stored in this metric.
   - The error of the MRS is calculated by measuring the distance between two points with a known distance between. This means that the error throughout the lab experiments has as many DOF as the number of tracked MRS points while this metric has as many DOF as the number of modal shapes. This difference in vector spaces complicates the comparison between the 100% metric and the error of the MRS, leaving us with no suitable metric which is independent of the wing's structure.

   We attempt to address the first issue with the second metric and the second issue with the third metric.

2. *20% metric.*
   The L2 error between the *the 20% most mobile* vertices of the point clouds created from $\boldsymbol{\xi}_i$ and $\hat{\boldsymbol{\xi}}_i$, normalized by the vertex count. To determine the most mobile vertices, we compute the average displacement over the entire displacement dataset, and pick the *20%* with the highest average value. The purpose of this metric is to mitigate the first issue described above by only examining a smaller, more informative subset of the vertices. This metric will be referred to as the *20% metric*.

3. *Maximal Vertex Distortion.*
   We define the Maximal Vertex distortion between $\boldsymbol{\xi}_i$ and $\hat{\boldsymbol{\xi}}_i$ to be the largest Euclidean distance between the true and estimated positions of a single vertex in the point clouds generated by $\boldsymbol{\xi}_i$ and $\hat{\boldsymbol{\xi}}_i$. Due to the second issue, we use this metric to provide an upper bound on the vertex distortion in a way independent of the wing's structure.

Throughout the experiments, all these metrics are calculated for each sample and averaged over the training set and over the validation set respectively.

3.3 Dataset Generation

The modal displacements can be described as a superposition of the *static modal displacements* $\{\boldsymbol{\xi}_i^s\}$ and *dynamic modal displacements* $\{\boldsymbol{\xi}_i^d\}$, as presented in equation 3.1 and they represent the physical range of values possible in flight.

$$\boldsymbol{\xi}_i = \boldsymbol{\xi}_i^s + \boldsymbol{\xi}_i^d \tag{3.1}$$

The static displacements depend on structural properties of the wing, their participation is amplified linearly with the *Angle of Attack* (α). The static modals displacements used in this experiment are presented in equation 3.2.

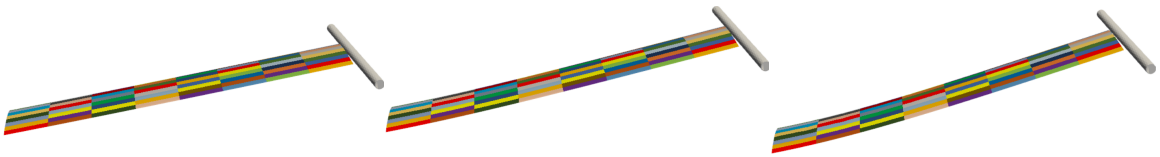
$$\{\xi_i^s\} \in \{\alpha \cdot \xi_{norm}^s | 1 \leq \alpha \leq 10,\ n \in \mathbb{N},\ \alpha = n \cdot 0.05\} \tag{3.2}$$

$$\xi_{norm}^s = \begin{matrix}[1.442\text{E}{-3},\ -7.104\text{E}{-5},\ -1.516\text{E}{-9},\ -4.152\text{E}{-5},\ -5.702\text{E}{-6},\\ 2.361\text{E}{-7},\ \ 1.554\text{E}{-6},\ \ 1.281\text{E}{-09},\ \ 1.333\text{E}{-8},\ \ \ 5.993\text{E}{-7}]\end{matrix} \tag{3.3}$$

The static modal displacement set we used contains 180 different α values, uniformly split in the range $[1, 10]$. This set was chosen to create a large number of static displacements while allowing them to visually differ from each other (an increase of 0.05 in α created a small visual difference in the static shape of the wing), thus reducing possible over-fitting on a single static shape. A sample of this set is visualized in figure 3.1. The dynamic displacements are the structural response of the wing to random gusts. A set of 16384 dynamic displacements was generated by the Aeroelasticity Lab in the Technion to emulate realistic flight conditions. Each of these dynamic displacements was paired with a single static displacement sampled from the set in equation 3.2. This creates a database of 16384 modal displacements $\{\xi_i\}$ linked with their appropriate structural displacements $\{u_i\}$, using equation 2.1. The structural displacements are added to the FE+ model point cloud to create a 3D mesh $\{u_i'\}$ representing a position the wing may achieve in flight. Using the PyVista [12] package these $\{u_i'\}$

Figure 3.1: A visualization of static modal displacements from equation 3.2 for different α values.

(a) $\alpha = 1$ *(b)* $\alpha = 5$ *(c)* $\alpha = 10$



are textured and photographed from a chosen camera position to create the final 2D images $\{\boldsymbol{X}_i\}$. The pairs $\{(\boldsymbol{X}_i, \boldsymbol{\xi}_i)\}$ are stored in a HDF5 [13] based database to be used by the DNN. The resolution of the images is 320x240, the highest resolution the Arducam camera is able to support natively while enabling the DNN to fit on this dataset within a reasonable time frame. We note that we may approximate our regression inference task as an image classification task by binning over the modal displacement distribution. In such tasks, it is commonplace to find datasets with up to approximately 100 samples per category. Leveraging this find, we treat the static shapes as related categories, and thus our database size was chosen to provide us with around 90 samples for each static shape the wing may achieve.

3.3.1 Textures

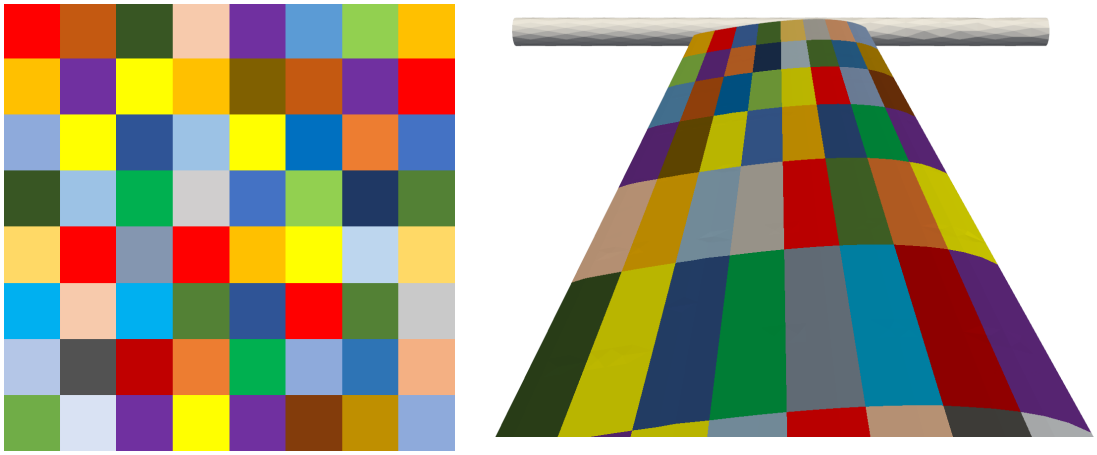
The texture images are mapped onto the wing in the following manner:

1. A 3D plane is chosen such that it minimizes the L2 error with the FE+ point cloud
2. A minimal rectangular area is chosen in this plane such that it includes the entire wing
3. The texture is resized and imprinted onto this rectangular area using planar projection

Figure 3.2 presents both a texture file and the resulting mapping and section 4.2 discusses the significance of these textures for inference.

Figure 3.2: Sample texture and its mapping onto the wing

(a) Rainbow Squares texture file *(b)* Rainbow Squares texture mapped onto the wing



3.4 Validation Split

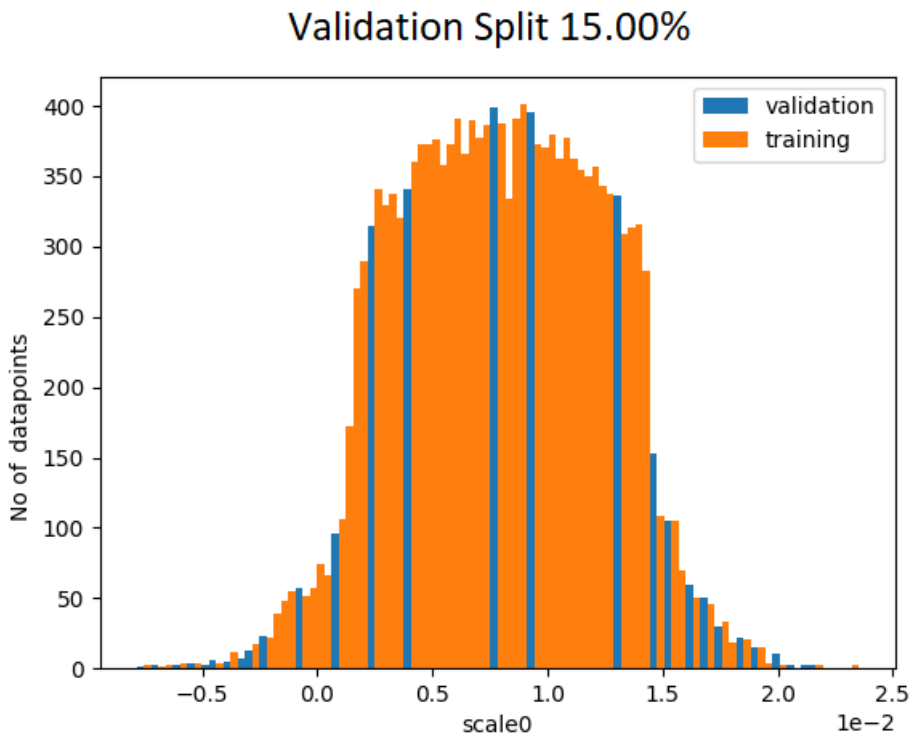
The validation set contains 15% of the overall data and was chosen by grouping the data points into 100 distinct groups based on the value of the first scale and choosing a set of groups that comprise 15% of our dataset to be our validation set. This split attempts to randomly select the validation set while dissuading similar samples from appearing in both sets.

Algorithm 1: Validation Set Selection

Require:
A data-set S, a set of disjoint groups of similar samples $\{g_i\}_{i=0}^{k-1}$ s.t. $S = \bigcup_i g_i$, the ratio between the sizes of the validation set and the whole data-set (λ), a step size (s) s.t. $\gcd(s,k) = 1$, a starting position (p) s.t. $1 \leq p \leq k-1$.
`/* This` s, p `choice guarantees that we iterate over all the groups before repeating */`
Result: A validation Set V
Initialization:
$\lambda = 0.15$, $k = 100$, $V = \emptyset$,
$\{g_i\}_{i=0}^{k-1}$ are chosen by uniformly splitting S based on the first modal displacement into k groups,
s, p are randomized.
while $|V| \leq \lambda |S|$ **do**
$V \leftarrow V \cup g_p$
$p \leftarrow (p+s) \mod k$
end
return V

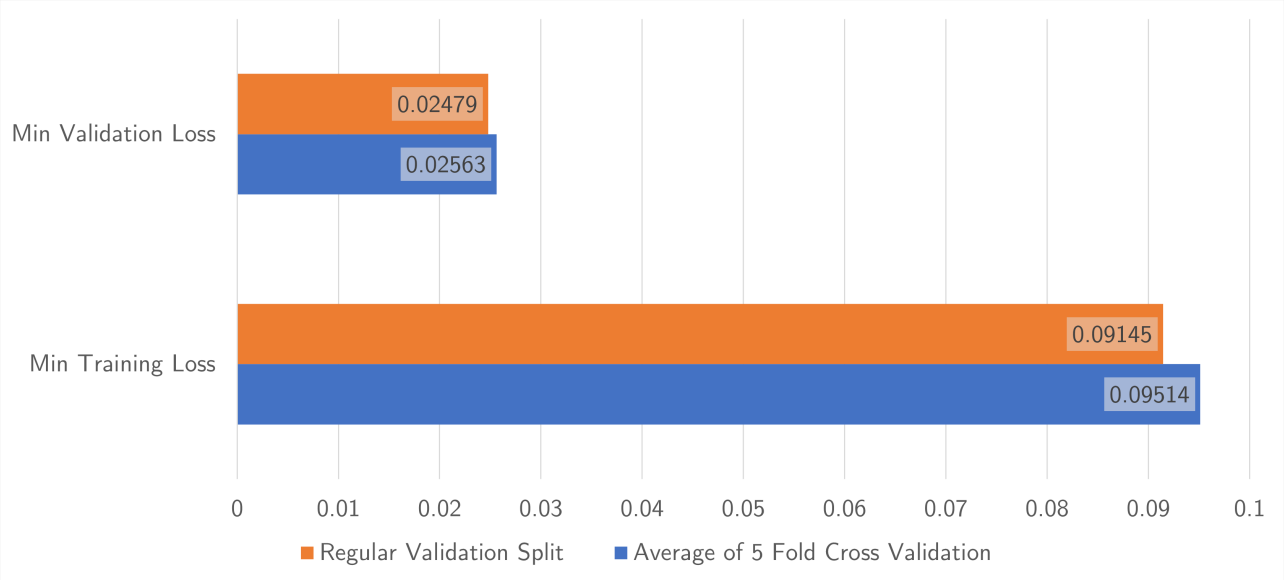
This choice was made using Algorithm 1. The rest of the samples were used for training. Figure 3.3 visualizes this choice. This specific validation set was chosen out of empirical considerations. We provide a comparison against the more commonly used 5-fold cross-validation split, relying on the DNN from section 3.6 below.

Figure 3.3: A histogram of the chosen validation split - the x axis define 100 groups in the range of the vales of the the first modal displacement and the y axis describes the amount of samples in each group



Comparison to 5-fold cross validation. While training the network we noticed that the validation loss of the network falls at some point below the training loss. This is an unexpected result and may imply a very poor choice of an "easy" validation set. We evaluate this loss in comparison to a 5-fold cross validation loss which provides a statistically reasonable evaluation of the DNN. We elected to use a validation set instead of using cross validation for all the experiments since cross validation significantly increases the experiment run time.

Figure 3.4: A comparison of the minimal losses achieved using our validation split and the average of the 5-fold cross validation. This experiment uses the Blue Squares texture apparent in figure 4.2, no noise, a batch size of 64, the Top Center camera apparent in figure 4.4 and a Black and White image type.



As we can see in figure 3.4, our validation split provides us with comparable results to the 5 folds cross validation both in the validation loss and in the training loss, substantiating the choice of this validation set. We acknowledge the anomaly of the validation loss being lower than the training loss. This may be due to the Batch Normalization [14] layers in the MobileNetV2 [15].

3.5 Noise

In order to place an upper bound on realistic noise that may appear in the experimental phase of the study we used three forms of noise:

1. Shot noise with $\lambda = 0.6$ as an upper bound for the sampling error of the Arducam OV7251 pictured in figure 2.3, the exact shot noise of the camera could not be determined easily, and so for simplicity we chose a high upper bound.

2. *Camera Shake*[1] - To simulate camera shake that may be caused due to the camera moving relative to the wing we skew the camera position, focal point, and up vector describing the camera. These coordinates were skewed by $\boldsymbol{\epsilon}_{1,i}, \boldsymbol{\epsilon}_{2,i}, \boldsymbol{\epsilon}_{3,i}$ where $\boldsymbol{\epsilon}_{1,i}, \boldsymbol{\epsilon}_{2,i}, \boldsymbol{\epsilon}_{3,i}$ were sampled from centered standard normal random vectors with a standard deviation of 0.0005, 0.001, 0.01 respectively.

3. *Backgrounds* - A random background chosen from a set of 15 distinct images taken in the wind tunnel. These simulate an additional form of noise that may appear under laboratory conditions. A sample of these is shown in figure 3.5.

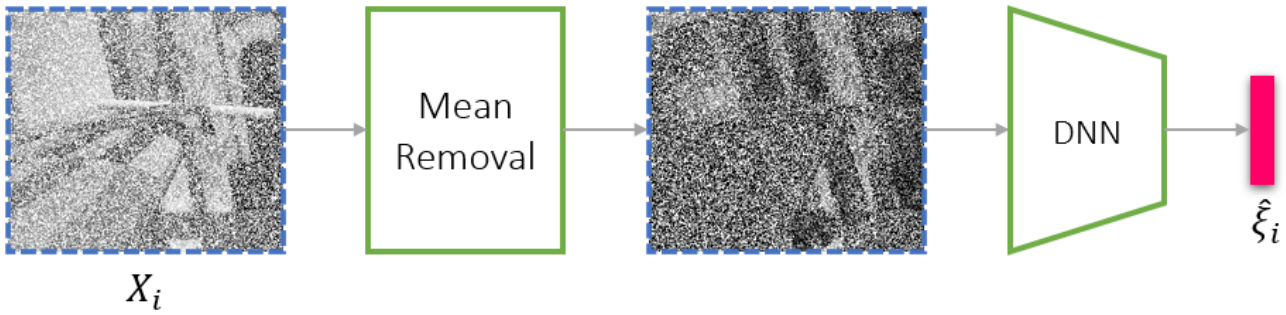
We will use these three forms of noise in the next chapters. They will be refereed to as *noisy conditions*.

[1] A visualization of the camera shake on a stationary wing https://youtu.be/c8f8ZfLrc2k

Figure 3.5: Sample backgrounds taken in the wind tunnel

3.6 DNN Overview

Figure 3.6: DNN Setup - A visualization of the network, receiving a noisy input ($\boldsymbol{X_i}$) as discussed in section 3.5, centralizing it by removing an image of the mean shape of the wing, and inferring the modal displacements ($\boldsymbol{\xi_i}$) using the DNN.



This setup, depicted in figure 3.6 uses a MobileNetV2 [15] based architecture as its DNN, the only difference being the depth of the first layer to accommodate different image formats when necessary. This architecture was chosen due to its established reliability, and its affinity to low-power edge devices, such as those that can fit onto light air vehicles. The network receives a monochromatic or RGB image ($\boldsymbol{X}_i$) and outputs the associated modal displacements $\hat{\boldsymbol{\xi}}_i$. The loss function used throughout the experiments is smooth L1 loss found empirically to work well by us, comparing the ground truth modal displacements with the ones estimated by the network. This function was first introduced in [16] for solving bounding box regression to great effect. For this function, we define $\beta = 1$. This is justified because of the arbitrary modification of $\boldsymbol{\xi}_i$ used for numerical stability, detailed further in section 3.6.1. The optimizer used is *Adam* [17] with a *Cosine Annealing learning rate scheduler* [18], cycling between a minimal learning rate of 10^{-5} and a maximal learning rate of 10^{-2} over 10 steps in a cosine-like manner (parameters found to work best in practice). This setup provided numerically stable errors while allowing us to focus on assessing the camera and texture configurations. Additionally, in order to speed up training, only a 2457-sized subset of the training set was used in each epoch, this subset was selected randomly in each epoch.

3.6.1 Numerical Stability

The values of $\boldsymbol{\xi}_i$ are in the range $[10^{-7}, 10^{-1}]$, thus when passing through the network we quickly encounter *arithmetic underflow*. In order to achieve meaningful results we use $10^4\boldsymbol{\xi}_i$ for fitting the network and then recover $\boldsymbol{\xi}$ from the output. This solves the numerical instabilities while providing loss values that benefit from the structure of the smooth L1 loss function.

3.6.2 Baseline

We define the baseline configuration to be the best configuration found throughout the DNN experiments and we will present the results of this configuration, trained under noise-less conditions as a reference point, acting as the lowest errors achievable in the experimental study. This configuration uses the Rainbow Squares texture apparent in figure 4.2, no noise, a batch size of 64, the Top Left camera apparent in figure 4.4 and a Black and White image type.

Chapter 4

Analysis

4.1 Theoretical Frequency Range

We attempt to find the maximal theoretical spectral range recoverable from 2D projections by assessing the contribution of each modal shape to the calculation of the modal displacement. We denote the maximal pairwise L2 error between two shapes in our database to be δ_{max}, and define the error of approximation of a deformed wing from the dataset, u_j using only the first i modal shapes to be:

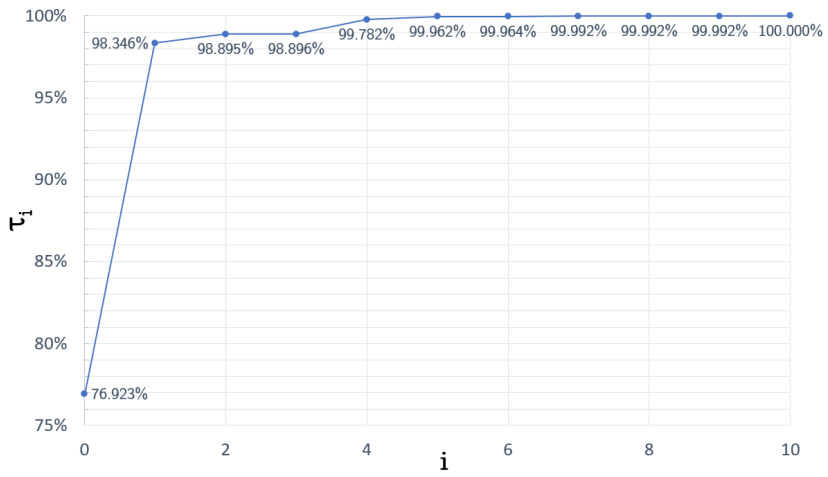
$$\delta_{i,j} = \left\| u_j - \sum_{k=1}^{i} \Phi_k \xi_k \right\|_2 \tag{4.1}$$

We further define the absolute accuracy of the approximation using just the first i modes over the entire dataset D as:

$$\tau_i = 1 - \frac{1}{|D|} \sum_{u_j} \frac{\delta_{i,j}}{\delta_{max}} \tag{4.2}$$

The absolute accuracy of the first 10 modes is presented in figure 4.1. As we can see, using no modal shapes at all (corresponding to an undeformed wing) provides us with a substantial accuracy of 76.923%, this is because dynamic motion of the wing is rather subtle in realistic conditions.
Note that by using just the first modal shape we may garner an accuracy of 98.346% with subsequent modal shapes, especially Φ_4 increasing the accuracy further to 99.782%. One should note that Φ_1, Φ_4 are the *bending* modes, so this result is expected.
In the subsequent section we will attempt to infer using the entire set of 10 modal shapes since the targeted set of modal displacements can always be reduced, lowering the complexity of the task and potentially improving the performance of the network.

Figure 4.1: The absolute accuracy of the approximation (τ_i) as a function of the number of modal displacements used (i)



4.2 Texture Optimization

In this section we propose seven distinct textures for the wing and test them under *noisy conditions*. The rest of the configuration is unchanged from the baseline. The seven pictures, also shown in figure 4.2 are:

1. *Black*
2. *Red*
3. *Blue Squares*
4. *Rainbow Squares*
5. *Red-Black Stripes*
6. *Constant Gaussian*
7. *Changing Gaussian (new Gaussian texture in each sample)*

The monochromatic, Red and Black textures were tested as they are quite simple to implement in a lab setting. The Rainbow Squares texture serves the purpose of placing clear geometric landmarks in the spatial and color spaces, in hopes of encoding additional geometric information in each image relevant for neural inference of spatial distances or alternatively, wing geometry. With this approach, we hypothesize that mapping to the require modal displacements will be achieved more readily.
The Blue Squares texture offers a slightly modified configuration, using only two colors. The Red-Black Stripes texture was used to test a completely different type of geometric pattern incorporating relatively large shapes, and examines the inference task when the texture suffers from spatial aliasing.
The Constant Gaussian texture was used to assess the viability of textures with tiny patterns and the Changing Gaussian texture was used to test the performance of a network where no meaningful information is encoded in the texture.
An evaluation of the network where all input images in both the training set and validation set are blank (letting the network train and evaluate when no significant input is given) is also present to emphasize the terrible performance of some of the textures. A comparison of the different textures is presented in figure 4.3, it presents a few interesting observations:

1. The Changing Gaussian texture performs significantly better than expected even though no information is encoded in the texture itself. This implies that inference is possible as long as the texture is distinct from the rest of the image.

2. The monochromatic textures achieve significant over-fitting, with the black texture performing worse than when the network receives no significant input. This may occur due to the input centralization mentioned in section 3.6. The centralization process subtracts a constant image ($\boldsymbol{X}_\mu$) from every sample, and thus pixels in the processed image may approach a constant (e.g. 0 for black) in two distinct cases:

   - The pixel is part of the wing and the respective value in ($\boldsymbol{X}_\mu$) is small.
   - The pixel is not part of the wing however subtracting its respective value in ($\boldsymbol{X}_\mu$) results in the color of the wing.

   These distinct groups of pixels are both mapped to the color of the wing, complicating inference, and resulting in over-fitting.

3. The Rainbow Texture achieving the best results w.r.t. the validation metrics, followed closely by all non-monochromatic textures. This implies that most textures with either notable geometric landmarks (such as large squares or stripes) or a vast color palette are viable, while a combination of these achieves the best results.

4. All four graphs in figure 4.3 look similar to one another, implying a high degree of correlation between the loss function and all the metrics, and justifying their choice despite the flaws mentioned in section 3.2.

Figure 4.2: The different textures used in the comparison - The top photos are taken without noise, the bottom photos are the noisy image as they are presented to the network for inference.

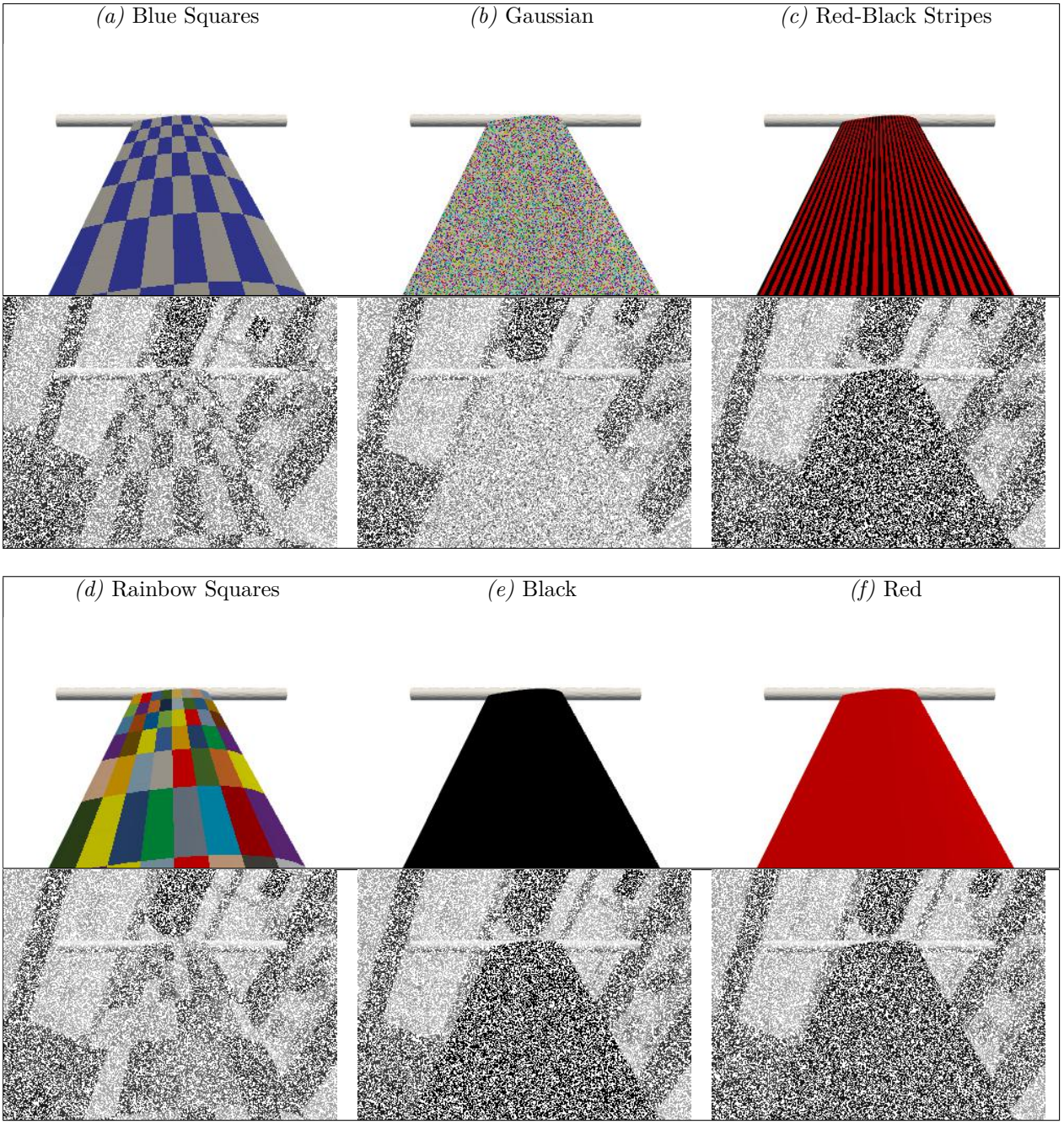
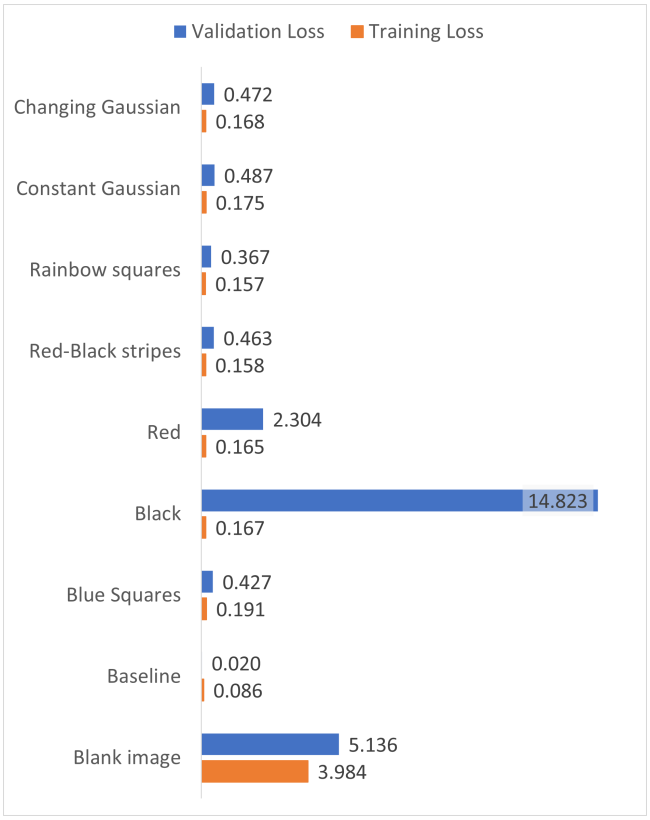
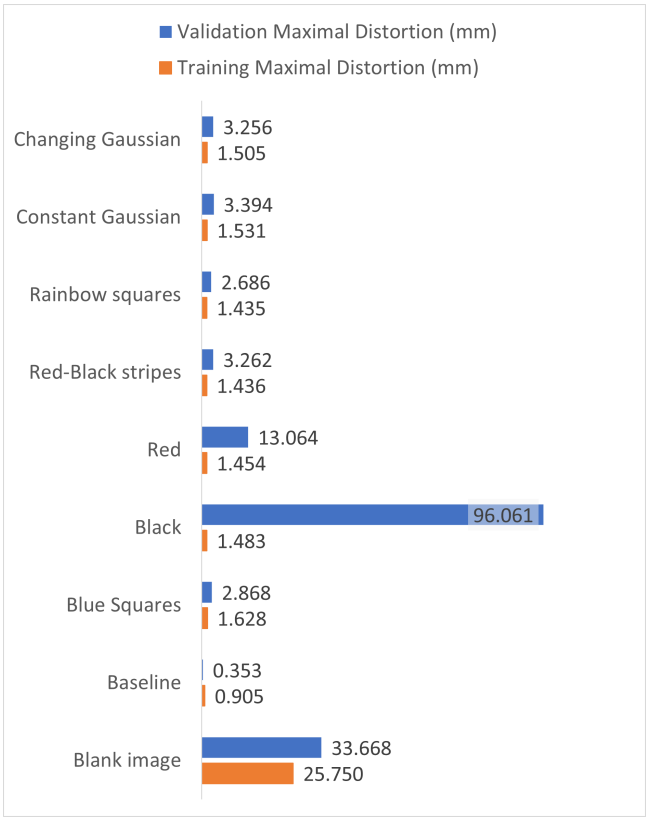


Figure 4.3: The minimal errors achieved throughout training, comparing the benefits of different textures apparent in figure 4.2. This comparison was done with the Top Center camera from figure 4.4, a batch size of 64, Black and White image type, and *noisy conditions* as discussed in section 3.5.

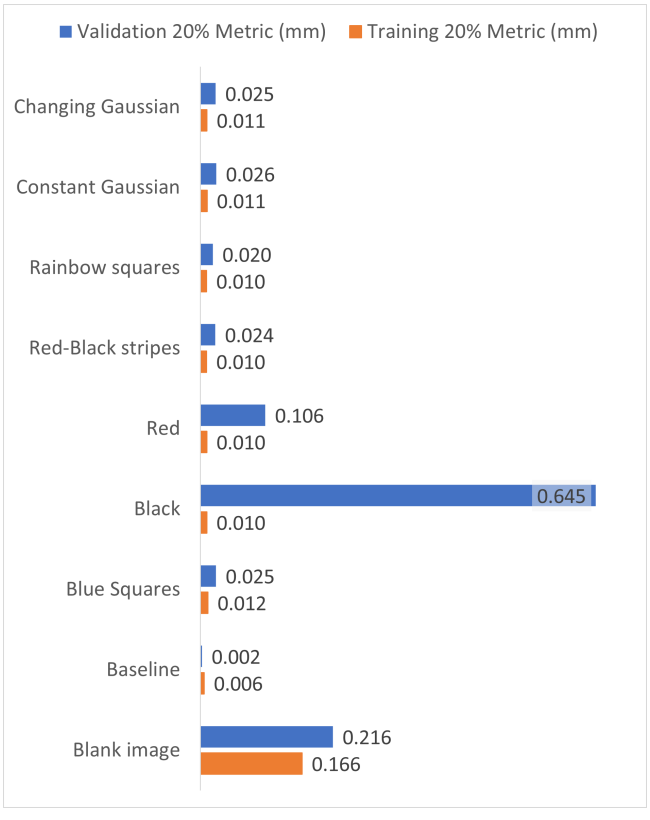
(a) Loss



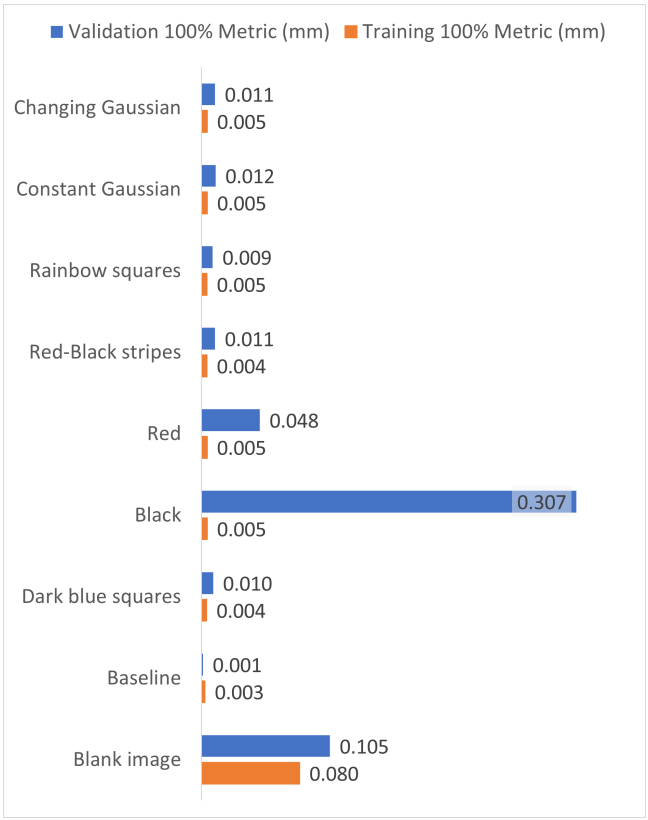
(b) Maximal Vertex Distortion



(c) 20% Metric



(d) 100% Metric

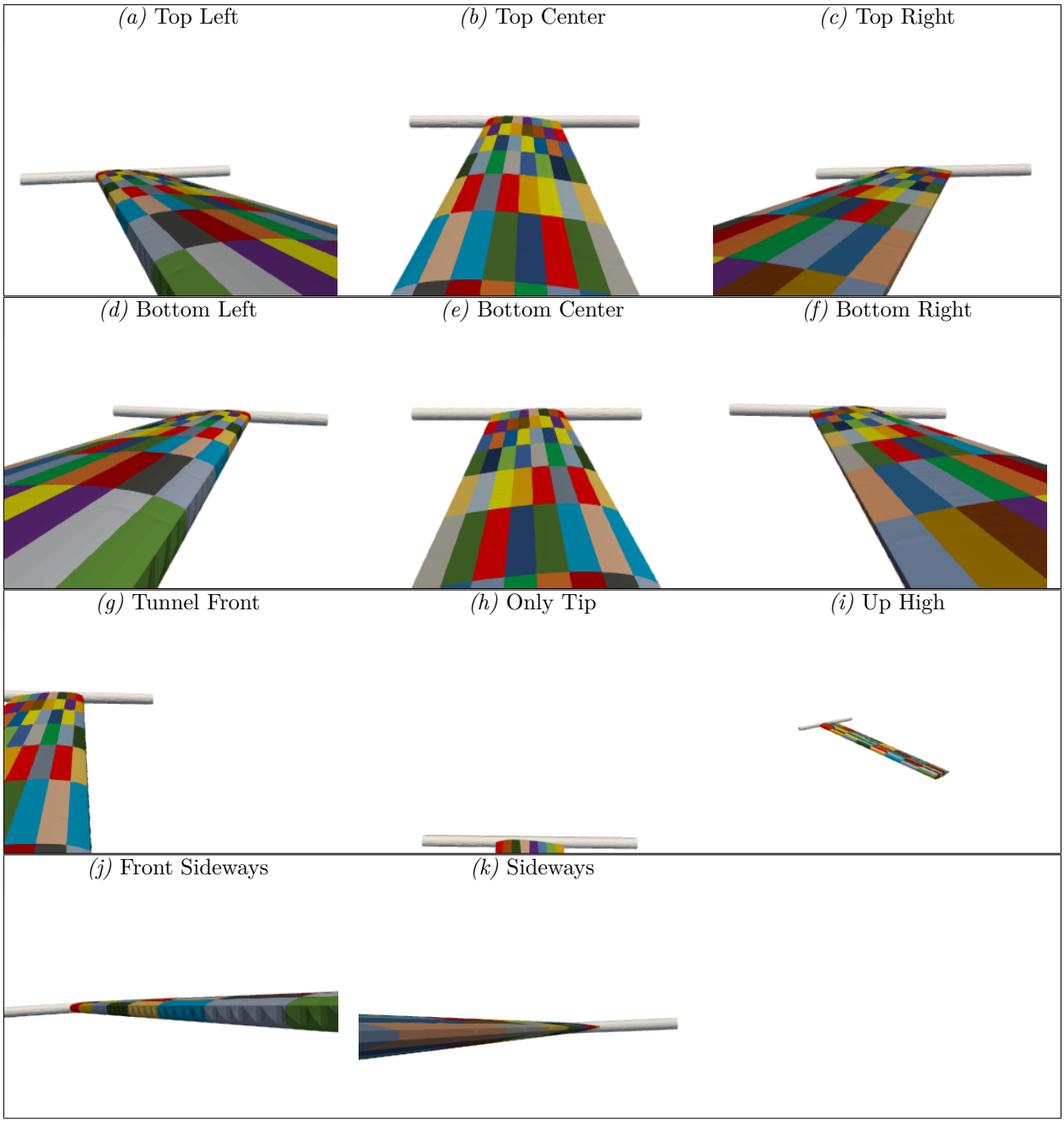


4.3 Camera Optimization

4.3.1 Camera coordinates optimization

In this section we compare eleven distinct camera angles. These angles, pictured in figure 4.4 aim to simulate different possible camera placements on an airplane. In addition, the Up High and Only Tip angles which provide a lacking view of the wing attempt to answer the question of *how bad is a bad camera angle?* The results of this comparison are available in figure 4.5, followed by an analysis.

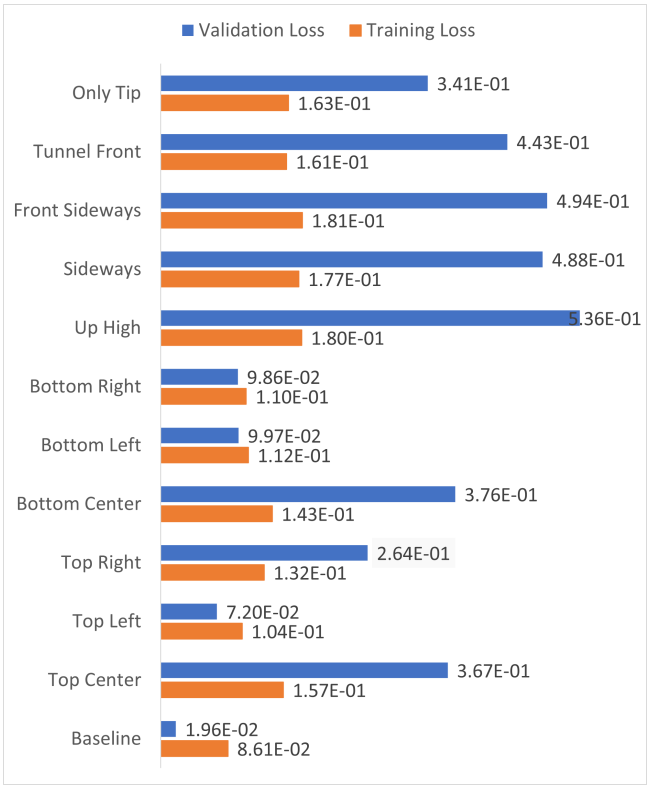
Figure 4.4: A visualization of the different camera angles used, pictured is the FE+ model with the Rainbow Squares texture with no displacement



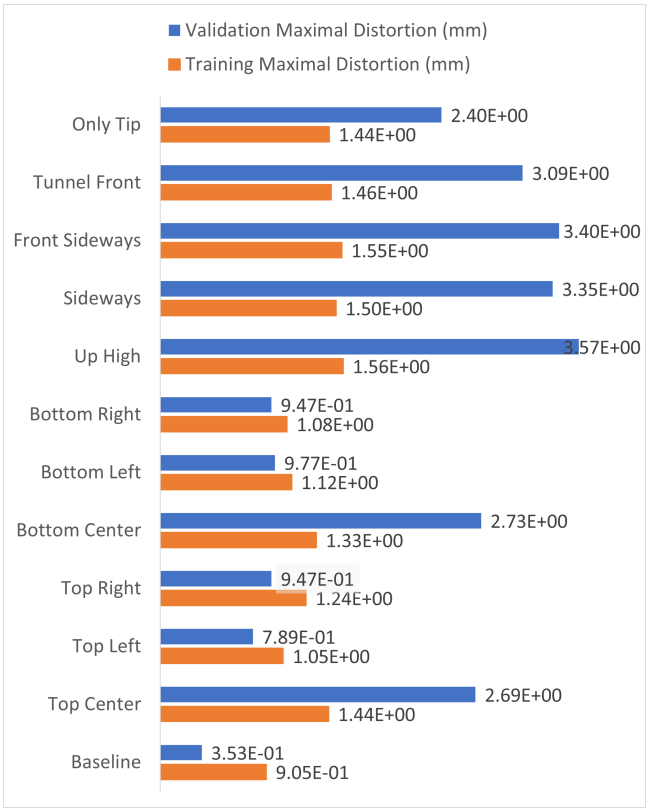
We will begin by discussing the six "main" cameras, Top Left, Top Center, Top Middle, Bottom Left, Bottom Center, Bottom Right. These cameras capture similarly sized parts of the wing, larger than the rest of the cameras, and we expected them to provide the best results. The best performing cameras are in this set. They are Top Left, Bottom Right, and Bottom Left in this order. The superiority of the Top Left camera over the Bottom Right and Bottom Left cameras can be explained due to the static displacement of the wing. In our dataset the wing curves upwards, providing a clear view of the curved wing for the top cameras and a slightly obscured one for the bottom cameras.

Figure 4.5: The minimal errors achieved throughout training, comparing the benefits of different camera angles apparent in figure 4.4. This comparison was done with the Rainbow Squares Texture, a batch size of 64, Black and White image type, *noisy conditions* as discussed in section 3.5 and different camera angles.

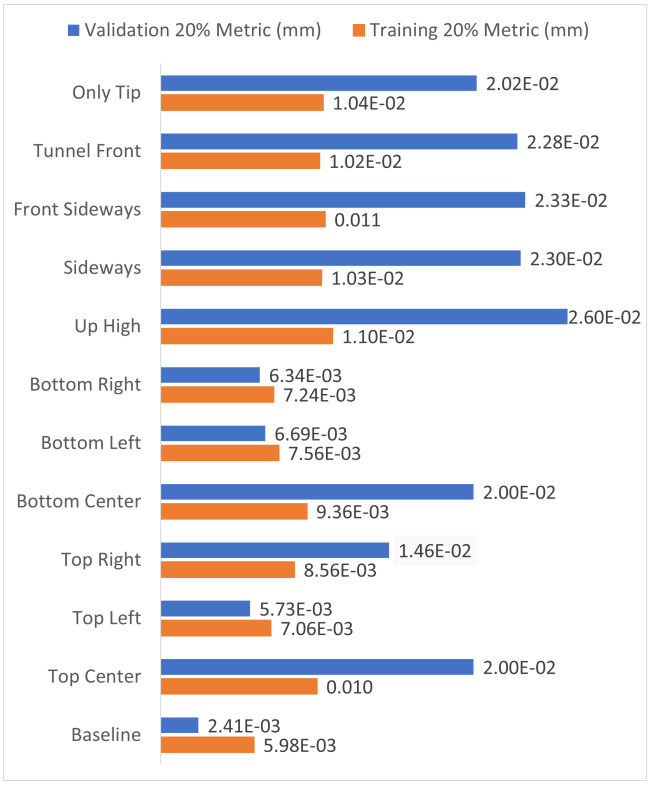
(a) Loss



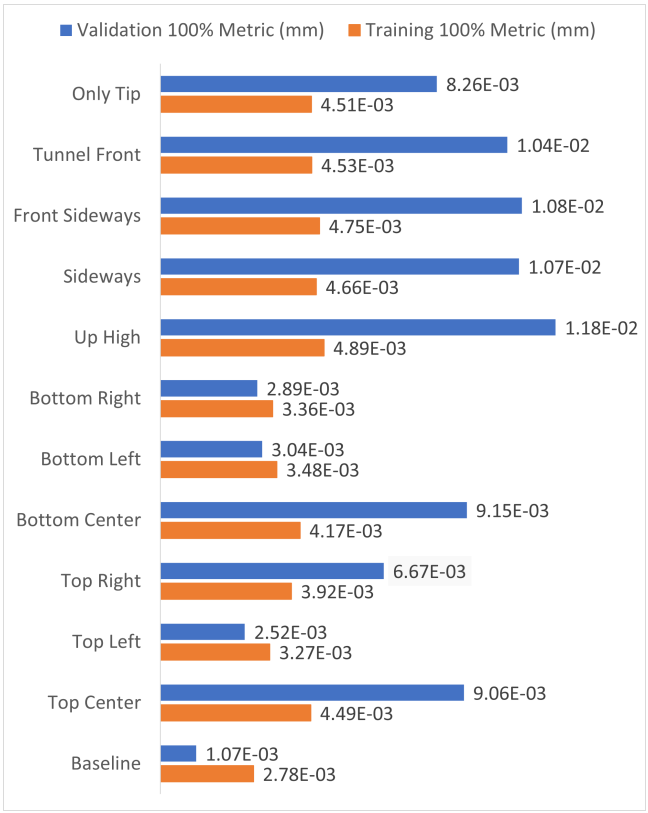
(b) Maximal Vertex Distortion



(c) 20% Metric



(d) 100% Metric



The next best-performing configuration is the the Top Right camera. The validation metrics of this camera are notably worse than the results of the previous camera. The cause of this phenomenon is unclear. The worst performing configurations out of these six cameras are achieved with the Top Center and Bottom Center cameras, this may be due to their lacking view of the side of the wing, thus limiting their understanding of the curvature or other geometric notions apparent in each photo.
We will discuss the rest of the cameras in order, from best performance w.r.t the validation metrics. Perhaps the most interesting result is achieved by the Only Tip camera, which had a validation performance similar to that of the Top Center, Bottom Center, and Tunnel Front cameras even though its images showcase only a tiny part of the wing. One could interpret this find by declaring the wing tip and the Only Tip camera angle as being geometrically informative, and similar in their information content to the two other camera angles.

Recall that most of the wing movement may be explained by the first two modal displacements, as shown in 4.1. The vertical displacement of the wing tip is determined mostly by the first mode, and its angular displacement determined mostly by the second, as may be seen in figure 2.1. After a linear projection on to the 2D image plane and discretization, these may be realized as simple geometric quantities in the image, such as the vertical pixel offset of the tip from the bottom of the photo or the tip's angle as measured from the axis defined by the bottom of the photo. Based on the empirical findings, we hypothesize that once a neural model accurately deduces such quantities in the image, the inverse mapping back to modal displacements is more readily achieved. An interesting experiment might be devised by simply replacing the entire input image with just the first or the second set of the two geometric quantities mentioned, but we leave this for future work. We further note that the exact significance of the shape of the tip itself has also not been researched, nor have we examined the parts of the various images that best explain the performance (and their correlation to the most dynamic vertices), as may be realized with neural explanation and visualization methods.

Next in the ordering are the Sideways, Front Sideways and Up High cameras. The performance of the first two suggest a profile of the wing on its own is severely lacking in information. This is in contrast to the best performing cameras defined above, which clearly utilize the partial view of the wing's profile, achieving a better performance than cameras where this view is not available. The performance of the last, the Up High camera, is also expected since the dynamic parts of the various images in this angle are significantly smaller.

We note that the performance of all configurations above are useful in general, and achieve validation results that are an order of magnitude smaller than the results achieved when no significant input is passed to the network (Blank Image experiment from section 4.2). Additionally, figure 4.5 demonstrates once more the high correlation between all the different metrics.

4.3.2 Channel Information Optimization

In this section we compare different input image formats in order to determine the viability of each format. The four formats are:

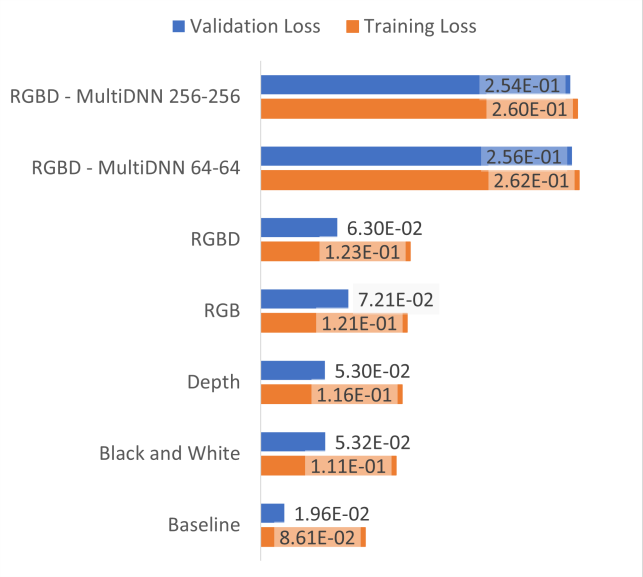
1. Black and White (BW)
2. Depth
3. RGB
4. RGBD

In each of the single DNN experiments (BW, RGB, RGBD, Depth), the networks input width is modified to accommodate the input format, (width of 1 for BW, Depth; width 3 for RGB, width 4 for RGBD). We recognize that in the RGBD experiment, the combination of Depth and Color channels blends inputs from completely different spaces and we expect this to provide incoherent results, thus two additional multi-modal networks (denoted as *Multi-DNNs*) were tested for inferring from RGBD data. In both networks, a separate DNN was used for the RGB data and Depth data, each one outputting a vector of size 128. These two vectors were concatenated and passed to a dense-layer network with 3 dense layers, with Relus between them (One experiment uses 64-sized dense layers and the other 256-sized layers). These configurations were chosen due to the proven viability of latent-based solutions for combining multiple input formats. The two sizes were used to assess multiple different combinations of the latent data, however better methods should be researched further.

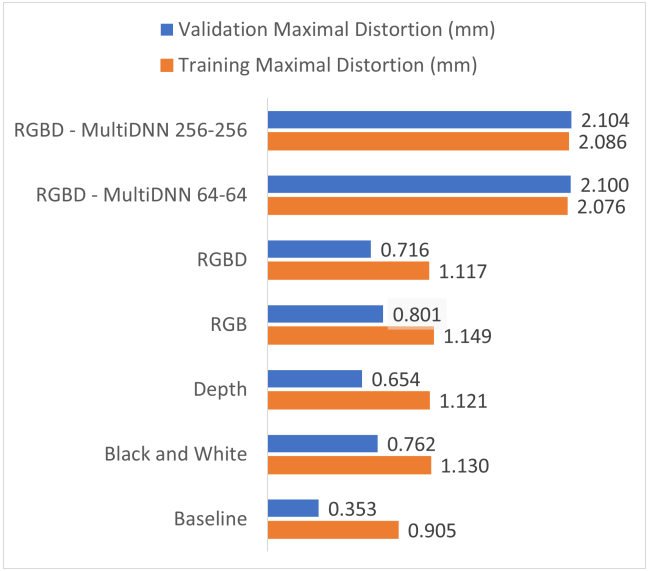
As we can see in figure 4.6, unlike the previous experiments, none of the configurations significantly outperform the others. We note that using the RGB input format provides no advantage over using a simpler, black and white format. We also note that the black and white format is worse in some metrics than the depth format.
The RGB/RGBD configurations don't outperform the Black and White/Depth configurations. A possible explanation is that the added information in these formats isn't compensating for the additional complexity of the network caused by the increase in input dimensionality. However, the RGBD configuration did slightly outperform the BW and RGB configuration, implying that even though this sort of combination may be misguided,

Figure 4.6: The minimal errors achieved throughout training, comparing the benefits of different input image channels. This comparison was done with the Rainbow Squares Texture, a batch size of 64, Top Left camera angle, camera shake and realistic backgrounds, and different input image channels.

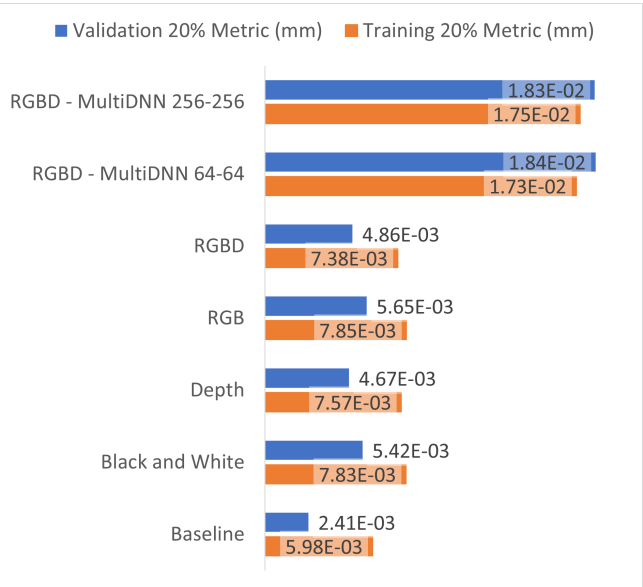
(a) Loss



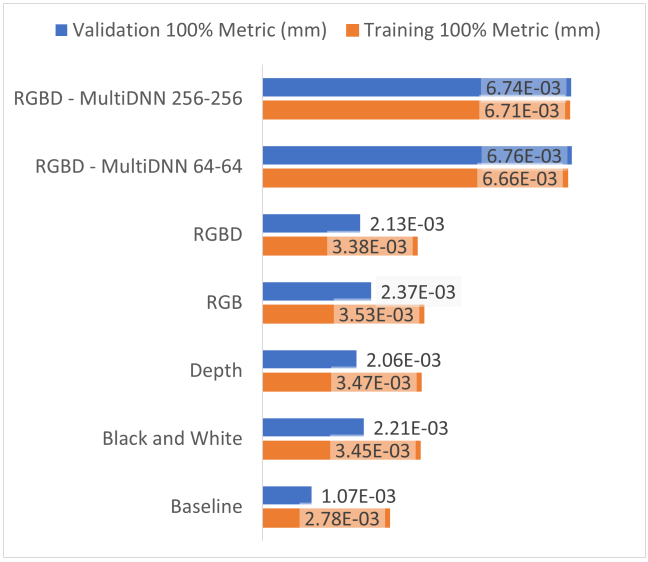
(b) Maximal Vertex Distortion



(c) 20% Metric



(d) 100% Metric



the naively added depth information does compensate for the added complexity. The Multi-DNN networks perform poorly, possibly attributed to the method of processing the latent layer, or to a poor ability of the MobileNetV2 to extract latent features from the inputs. Further research is necessary to determine the cause of these poor performances, and to find a better method of combining the color and depth inputs.

4.3.3 Camera count optimization

We turn to question the marginal performance gain achievable by aggregating multiple camera viewpoints as input, from two or more temporally synchronized camera sources. From a technical viewpoint, we should first question how to best aggregate these geometrically different input sources, as they vastly differ spatially. A direct concatenation of them as input channels is therefore ill advised, as discussed in the previous section.

Comparison of multi-modal architectures

We offer six different architectures, in order to asses the technical question raised above. To test the performance of the various systems, we chose to input images from six concurrent primary camera sources depicted in 4.4 (Top Left, Top Center, Top Right, Bottom Left, Bottom Center and Bottom Right viewpoints). All experiments were realized with the Rainbow Squares texture, realistic noise, a batch size of 12-16 (limited by vram) and a BW image format. We detail our architecture comparison in the following:

1. Stacked - Naively stacking the six input images on top of each other and passing them to the the network described in section 3.6.

2. Averaging - Using six MobileNetV2 based networks as described in section 3.6 and averaging their outputs.

3. Convex Combination - A neural ensemble approach. We use six MobileNetV2 based networks as described in section 3.6 and output a learned convex combination of their outputs. The weighting coefficients of the combination are learned by invoking two dense-relu (64-64 size) layers, and a 64 linear-softmax on the outputs of the MobileNets as may be seen in Figure 4.7.

4. Frozen Convex Combination - Similar to the previous approach however the MobileNets were pre-trained with the best weights from the camera angle experiment and frozen, with the learning done only on weighting coefficients. The contrast between the two approaches serves to quantify whether pre-training approaches are viable in such a setting.

5. Latent Combination - An Encoder-Decoder approach for combining inputs, as seen in [19, 20]. Using six MobileNet based setups, each outputting a 128 sized image encoding. These outputs were concatenated to form a joint latent vector of size $128 \cdot 6$ and processed through two dense-relu (64-64) layers realized as a decoding or regressing process to produce the final result.

The results of these experiments are presented in figure 4.8. As we can see, surprisingly no method surpassed the single camera setup. The best configuration was the Convex combination. The poor performance of these configurations, especially of the proven Latent Combination configuration may be again due to the increase of input dimensionality. It is possible that each camera view point encodes more or less the same geometric information, and thus the gain in performance over a single camera viewpoint is marginal. This will be explored further in next subsection by only using a subset of the n-best performing cameras out of these six inputs on the best preforming configuration.

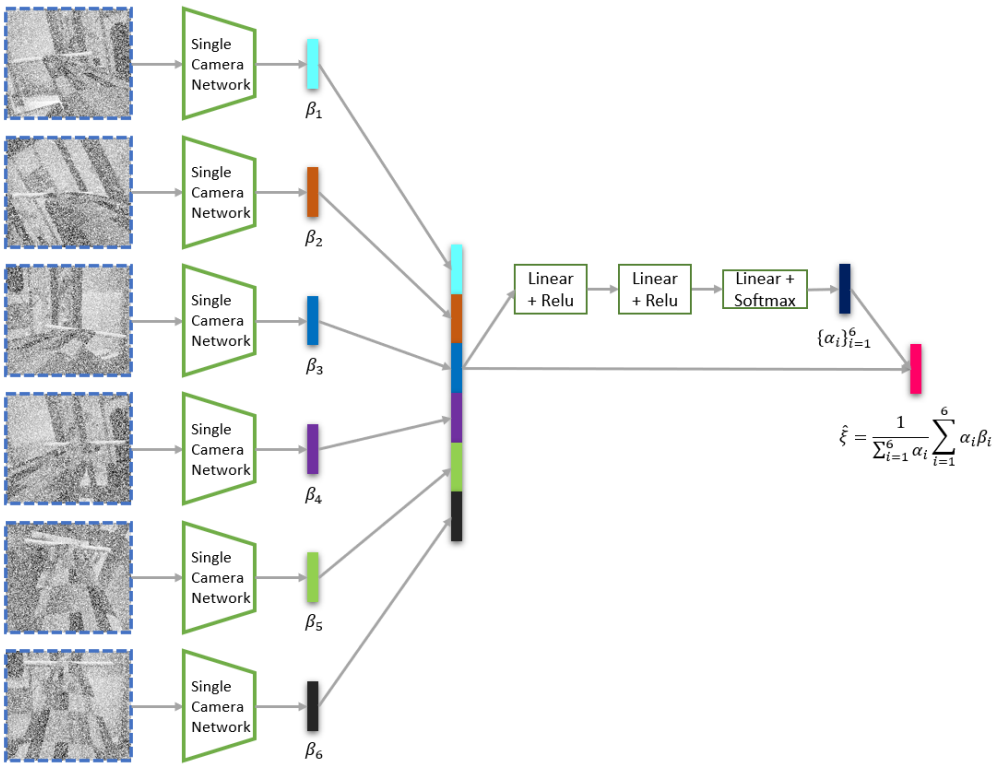
We further note that the Frozen Convex configuration performs significantly worse than the Convex configuration, which is commonly the case when optimizing over only a portion of the network weights. This might also imply that the latent space of these networks differs significantly from the space spanned by the outputs of all mono-camera setups. These are just a small sample of the different possible methods of combining multiple inputs and better results may be possible, which we leave for future work.

Effects of Camera Count

In this section we will examine the effects of changing the number of cameras used as input for the convex combination method from the previous section. In each experiment we input a subset of the n best performing cameras out of the six primary cameras (Top Left, Top Center, Top Right, Bottom Left, Bottom Center and Bottom Right). By using only the best performing cameras we hope to infer only using the more informative inputs, utilizing their distinct views and the distinct information each view holds. The results are summarized in figure 4.9.

As we can see, the lowest errors are achieved when using two inputs. Both the two camera and three camera setups slightly outperform the single camera setup, possibly attributed to the proper utilization of the information added by the new angles. The four, five, and six camera achieve higher errors, this may be due to the added inference complexity caused by the increase in dimensionality out-weighting the benefits of the less-informative inputs. We note that the worst performing setup is the five camera setup, possibly explained by an imbalance caused by using three cameras from a top perspective and two cameras from a bottom perspective, thus over-representing the shortcomings of the top perspective in some of the samples. This same increase is not apparent in the three-camera configuration. This may be due to third best camera results being comparable to

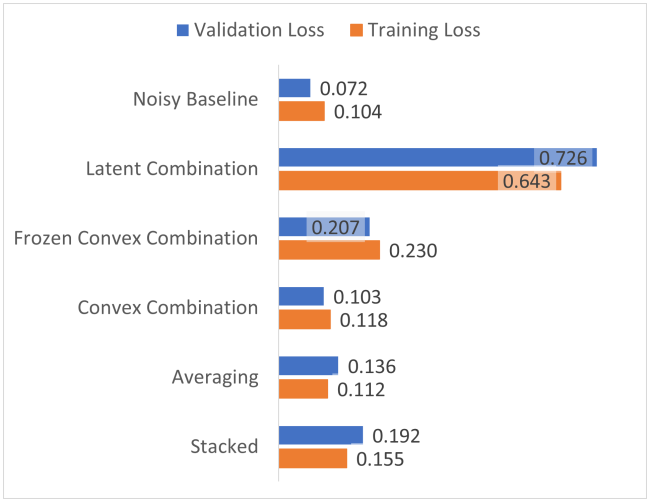
Figure 4.7: Multi Camera Convex Combination Network Setup - This setup utilizes multiple single camera networks, as described in 3.6 and achieved the best results out of the multi-DNN setups. The linear layers are of size 64.



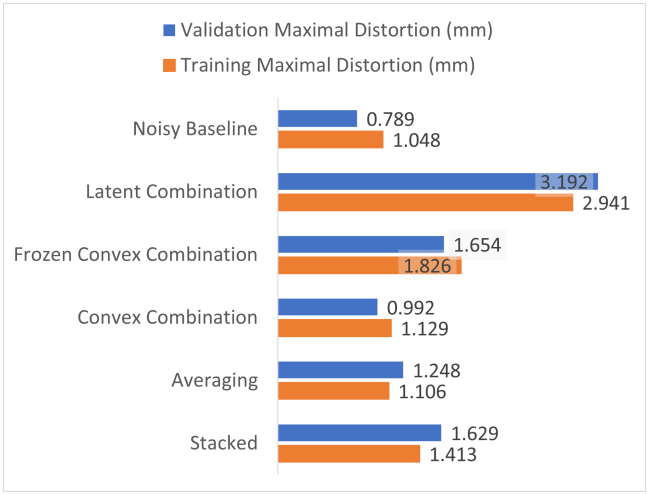
the best camera, while the fifth best camera errors are significantly higher. As we can see even the best of our multi-camera configurations didn't significantly outperform the single camera configuration, thus, unless better multi-camera networks become available, and with the added complexity of implementing such a multi-camera in a realistic environment, this approach seems disadvantageous under realistic conditions.

Figure 4.8: The minimal errors achieved throughout training, comparing the benefits of different multi-DNN networks. This comparison was done with the Rainbow Squares Texture, a batch size of 64, The six main camera angles, realistic noise, and BW input format. The best performing single camera setup (Noisy Baseline) is added for reference.

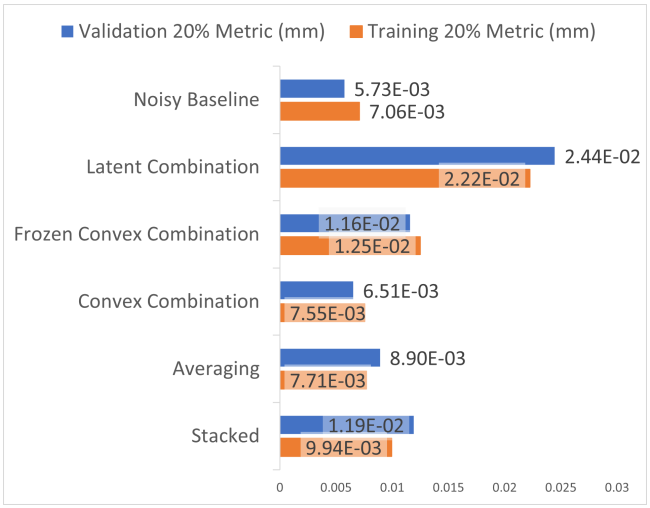
(a) Loss



(b) Maximal Vertex Distortion



(c) 20% Metric



(d) 100% Metric

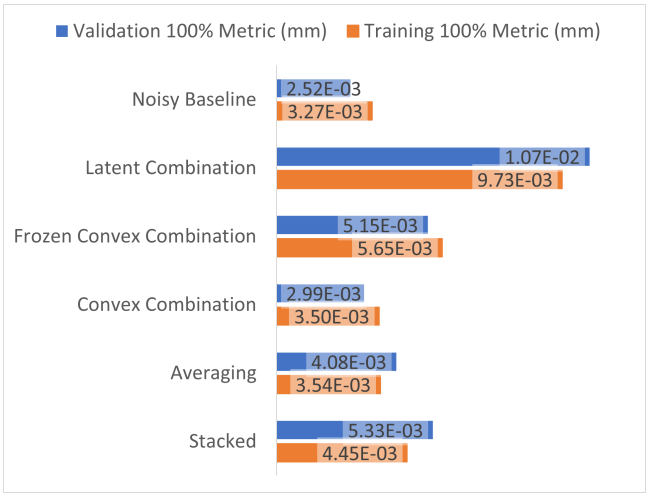
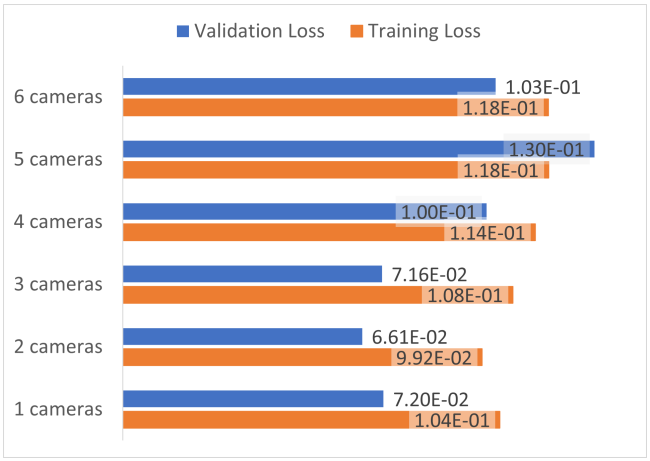
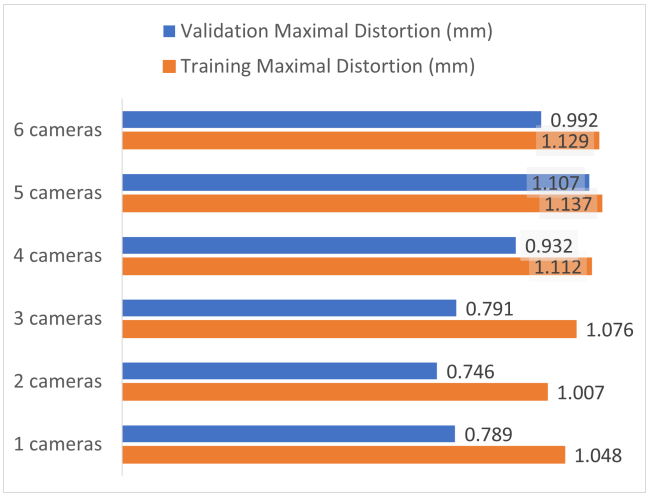


Figure 4.9: The minimal errors achieved throughout training, comparing the benefits of different camera amounts. This comparison was done with the Rainbow Squares Texture, a batch varying batch size of 16-64 (dependant on available VRAM), realistic noise, a BW input format, and a subset of 1-6 cameras used as input for the Convex Combination network from figure 4.7, 1 camera being the Noisy Baseline.

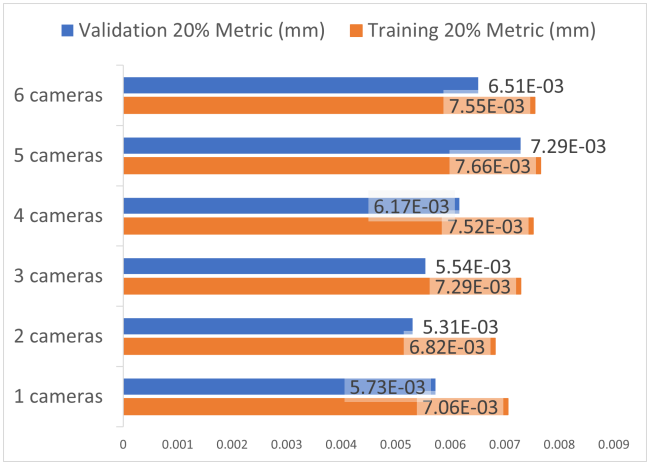
(a) Loss



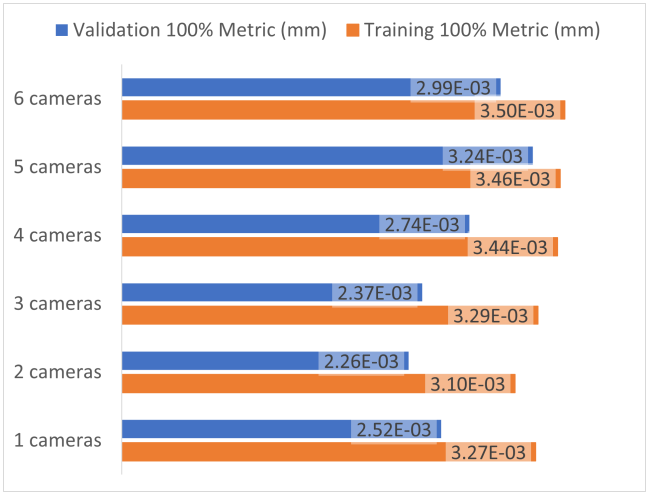
(b) Maximal Vertex Distortion



(c) 20% Metric



(d) 100% Metric



4.4 Theoretical Regression Error Under Lab Conditions

We recommend the following configuration for the lab conditions.

- The best texture according to section 4.2 is the Rainbow Squares texture and thus the wing is dressed accordingly.
- The best camera positions according to section 4.3 are the Top Left, and Bottom Right camera positions, providing similar results. Utilizing two or three temporally synchronized cameras placed in different locations has been shown to improve results marginally, and special care needs to be placed in assessing whether this possible performance gain is beneficial under realistic conditions.
- Both monochromatic and depth image types provided similar results, according to section 4.3.2, while monochromatic cameras are cheaper and simpler to work with, depth cameras are able to perform denoising tasks such as background removal easily. This may be useful in later stages.
  Both setups should be tested and we expect them to provide similar results.

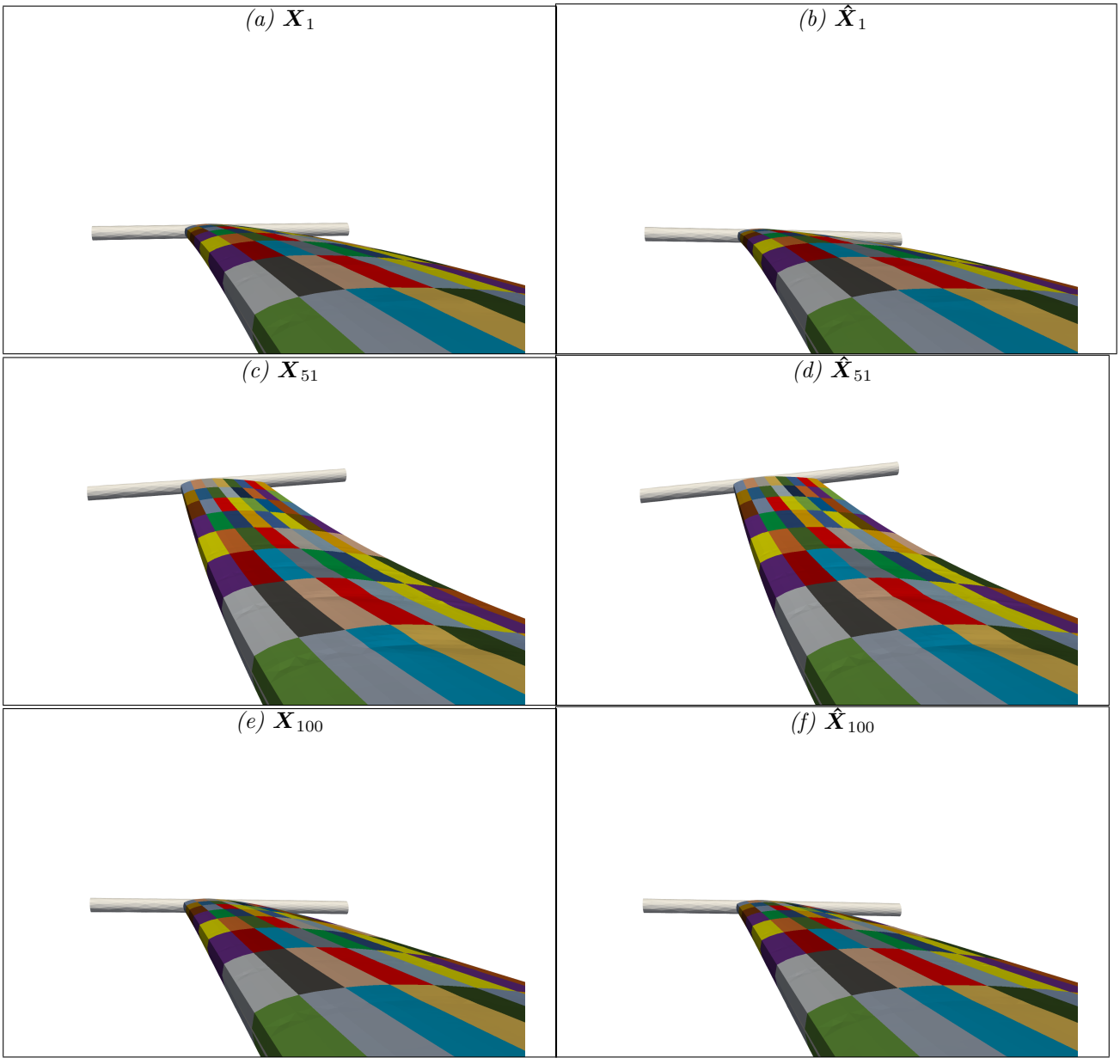
Under these conditions we expect the results under lab conditions to fall between the noiseless baseline results and the baseline configuration under realistic noise. This range is summarized in table 4.1.

Table 4.1: Expected Experiment Results Range - the minimal values are the results of the baseline and the maximal are the results of the baseline under *noisy conditions*.

	Training Loss	Validation Loss	Training 20 % Metric (mm)	Validation 20 % Metric (mm)	Training 100 % Metric (mm)	Validation 100 % Metric (mm)	Training Maximal Displacement (mm)	Validation Maximal Displacement (mm)
Minimum	8.61E-2	1.96E-2	5.98E-3	2.41E-3	2.78E-3	1.07E-3	0.9051	0.353
Maximum	0.104	0.072	7.06E-3	5.73E-3	3.27E-3	2.52E-3	1.048	0.789

A visualization of some of the worst predictions of the model, w.r.t the *100% metric* is presented in figure 4.10.

Figure 4.10: A visual comparison between a reconstruction using the true modal displacements $\boldsymbol{\xi}_i$ and their estimations using the DNN, $\hat{\boldsymbol{\xi}}_i$. The samples were taken using the best performing configuration, where $\boldsymbol{X_i}$, $\hat{\boldsymbol{X}}_i$ have the i-th worst reconstruction error over 100% of the vertices out of the validation set consisting of 2457 images.



Chapter 5

Conclusions and Further Research

5.1 Conclusions

In this study we have shown that the proposed DNN-based methodology for modal analysis performs well under computer simulations. We have also shown that a relatively small number of natural modes are able to provide a faithful reconstruction of the wing. Additionally, we have created and optimized a camera and texture configuration which will be used to experimentally test this methodology using a 3D printed wing in a wind tunnel experiment. We have also tested multiple multi-camera configurations, and managed to outperform the best single-camera network using the added information from these inputs, and shown that this increase is marginal at best.

5.2 Further Research

This study provides a foundation for the proposed DNN-based methodology however, further research will be necessary to mature it. We detail some interesting future research topics:

- *DNN Optimization* - No significant work was done into optimizing this network. Once experimental data becomes available, the DNN, including the prepossessing mechanism, should be adjusted to improve the latency and throughput of the network while maintaining a satisfactory error rate. This may enable real-time inference rates useful for a wing control system conditioned on the modal displacements.
- *DNN Specialization* - We recognize that for different aeroelastic applications, different natural modes, or sets of modes are necessary. Hence, different, more specialized DNNs may prove advantageous for different applications.
- *More Informative Metrics* - Each of the metrics proposed in this study had a disadvantage in some aspect. Other, more informative metrics may prove useful throughout the process of DNN optimization.
- *Accurate Wing Modeling* - The current visual distinction between the FE+ wing model and the CAD model discussed in section 2.3 will eventually impede us from enriching the experimental dataset using a synthetic one. Constructing an FE model which more closely resembles the CAD model may prove useful.
- *Multi Camera Setups* - In this study we have tested a very limited number of multi-input networks. These setups have not shown very promising results, and better configurations should be explored.

Chapter 6

Bibliography

[1] Livne, E., "Aircraft active flutter suppression: State of the art and technology maturation needs," *Journal of Aircraft*, Vol. 55, No. 1, 2018, pp. 410–452.

[2] Zeng, J., Moulin, B., De Callafon, R., and Brenner, M. J., "Adaptive feedforward control for gust load alleviation," *Journal of Guidance, control, and dynamics*, Vol. 33, No. 3, 2010, pp. 862–872.

[3] Pendleton, E. W., Bessette, D., Field, P. B., Miller, G. D., and Griffin, K. E., "Active Aeroelastic Wing Flight Research Program: Technical Program and Model Analytical Development," *Journal of Aircraft*, Vol. 37, No. 4, 2000, pp. 554–561.

[4] Zink, P. S., Mavris, D. N., and Raveh, D. E., "Maneuver Trim Optimization Techniques for Active Aeroelastic Wings," *Journal of Aircraft*, Vol. 38, No. 6, 2001, pp. 1139–1146, doi:10.2514/2.2884.

[5] Reich, G., Raveh, D., and Zink, P., "Application of active-aeroelastic-wing technology to a joined-wing sensorcraft," *Journal of Aircraft*, Vol. 41, No. 3.

[6] Yagil, L., Raveh, D. E., and Idan, M., "Deformation Control of Highly Flexible Aircraft in Trimmed Flight and Gust Encounter," *Journal of Aircraft*, Vol. 55, No. 2, 2018, pp. 829–840.

[7] Suh, P., Chin, A., and Mavris, D., "Virtual Deformation Control of the X-56A Model with Simulated Fiber Optic Sensors," Tech. rep., NASA, 2014. NASA/TM—2014–216616.

[8] Pak, C., "Wing Shape Sensing from Measured Strain," *AIAA Journal*, Vol. 54, No. 3, 2016, pp. 1068–1077.

[9] Gherlone, M., Cerracchio, P., and Mattone, M., "Shape sensing methods: Review and experimental comparison on a wing-shaped plate," *Progress in Aerospace Sciences*, Vol. 99, 2018, pp. 14–26.

[10] Freydin, M., Keren-Rattner, M., Raveh, D. E., Kressel, I., Davidi, R., and Tur, M., "Fiber-Optics-Based Aeroelastic Shape Sensing," *Vol. 57 No. 12, 2019 pp. 5094-5103 in the AIAA Journal.*

[11] Rao, S., *Vibration of Continuous Systems*, John Wiley and Sons, New Jersey, 2007.

[12] Sullivan, C. B. and Kaszynski, A., "PyVista: 3D plotting and mesh analysis through a streamlined interface for the Visualization Toolkit (VTK)," *Journal of Open Source Software*, Vol. 4, No. 37, 2019, p. 1450, doi: 10.21105/joss.01450.

[13] Collette, A., *Python and HDF5*, O'Reilly, 2013.

[14] Ioffe, S. and Szegedy, C., "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," , 2015.

[15] Sandler, M., Howard, A. G., Zhu, M., Zhmoginov, A., and Chen, L., "Inverted Residuals and Linear Bottlenecks: Mobile Networks for Classification, Detection and Segmentation," *CoRR*, Vol. abs/1801.04381.

[16] Girshick, R., "Fast R-CNN," in "Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)," IEEE Computer Society, USA, ICCV '15, 2015, p. 1440–1448, doi:10.1109/ICCV.2015.169.

[17] Kingma, D. P. and Ba, J., "Adam: A Method for Stochastic Optimization," , 2017.

[18] Loshchilov, I. and Hutter, F., "SGDR: Stochastic Gradient Descent with Warm Restarts," , 2017.

[19] Yuan, Z., Jiang, Y., Li, J., and Huang, H., "Hybrid-DNNs: Hybrid Deep Neural Networks for Mixed Inputs," , 2020.

[20] Wang, M., "Multi-path Convolutional Neural Networks for Complex Image Classification," , 2015.

[21] Cignoni, P., Callieri, M., Corsini, M., Dellepiane, M., Ganovelli, F., and Ranzuglia, G., "Meshlab: an open-source mesh processing tool." in "Eurographics Italian chapter conference," Salerno, Italy, Vol. 2008, 2008, pp. 129–136.

[22] Bernardini, F., Mittleman, J., Rushmeier, H., Silva, C., and Taubin, G., "The ball-pivoting algorithm for surface reconstruction," *IEEE Transactions on Visualization and Computer Graphics*, Vol. 5, No. 4, 1999, pp. 349–359, doi:10.1109/2945.817351.

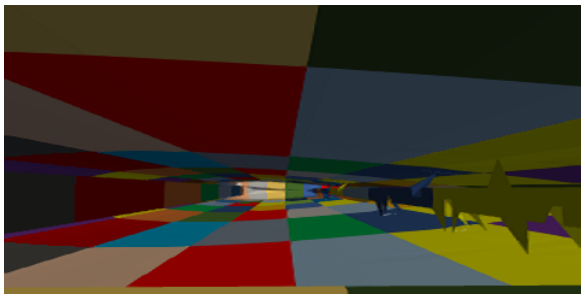
Appendix A

CAD Model approximation using the FE Model

The purpose of this section is to present the process in which we generate the FE+ wing model discussed in section 2.3.
We use the FE model and modify it using Meshlab [21], applying the Ball Pivoting algorithm [22] to construct the faces of the FE+ model hull.
The resulting hull had 9013 vertices and 16859 faces. The point-cloud vertices creating the wing's tip were seated on a one-dimensional line in the FE model and therefore did not create a visually accurate tip, so a cylinder-like shape, created by generating a circle around each FE tip vertex and building a convex hull around this shape is used as the tip. The resulting tip had 930 vertices and 906 faces. The FE+ hull and the FE+ tip are combined to create the Unpruned FE+ Model with 9943 vertices and 17765 faces. The Unpruned FE+ Model includes non-visible elements, these elements are removed to increase processing performance and to improve the quality of the texture mapping resulting in a leaner model with only 8484 vertices and 15960 faces. This resulting mesh can be animated using data from Finite Element Analysis as the hull of the FE+ model consists of a subset of the vertices of the FE model and each vertex in the tip can be moved according to the center vertex of its respective circle, these center vertices appear in the FE model. This process is visualized in figure A.2.

Figure A.1: A visualization of the FE+ model pruning done

(a) Unpruned FE+ Model Interior



(b) FE+ Model Interior

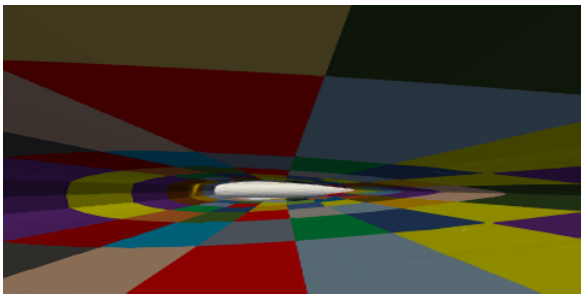
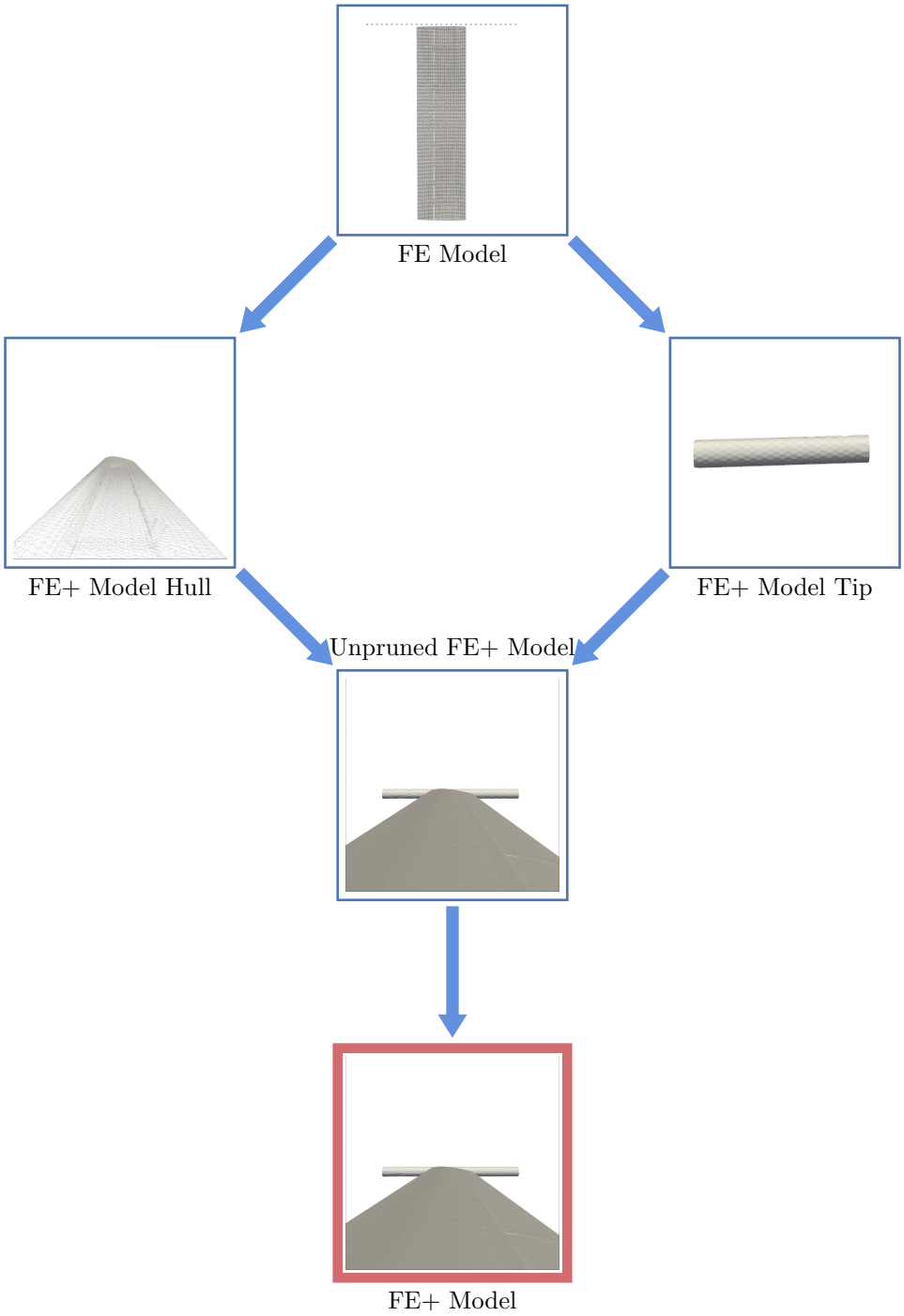


Figure A.2: FE+ Model Generation - Using Ball Pivoting [22] we create the hull of the FE+ model, the tip is generated by expanding each FE tip vertex into 30 different vertices forming a circle around each vertex, connecting these circles and forming a cylinder-like tip. The tip and the hull are concatenated to form the Unpruned FE+ model. We optimize this model by removing non-visible vertices to improve the performance of the database generation and create the final FE+ model.



Appendix B

MRS Analysis - A Note For The Experimental Phase

In this section we attempt to assess the projection of the MRS sampling error (The $100\%metric$ in mm) on the space spanned by the first 10 modal shapes $\{\Phi\}_{k=0}^{9}$.
The sampling error of the MRS can be modeled as additive Gaussian noise on the z-axis (perpendicular to the span of the wing) to each vertex of the point-cloud with a distribution $\mathcal{N}(0, 0.15^2)$, thus the expectation of the absolute value of this error is $E(\delta_{ir}) = 0.1192$, according to empiric tests in the Aeroelasticity Lab of the Technion. However, using a database of 16484 spacial displacements representing the shape of the wing, the average error of the *100% metric* between these samples and the mean displacement (calculated over the same database) is $E(\delta_{mean\_shape}) = 0.105$.

This error is lower than the expected MRS sampling error, this is an unclear result considering the modal analysis done using the MRS provides us with a meaningful model of the wing and the mean shape of the wing doesn't, we will attempt to explain this.

Modeling

We denote the projection of this error onto $\{\Phi\}_{k=0}^{9}$ as δ. Let us look at the modal displacement errors of δ, δ_{ir}, denoted as $\hat{\xi}$, ξ_{ir} respectively. Let $\Phi_{3\times n\times 10}$ be the transformation tensor from the modal space to the vertex space, $\Phi_{z(n\times 10)}$ be the matrix associated with the z-axis and Φ_z^+ be its pseudo inverse.
Let us express $\hat{\xi}$ using equation 2.1:

$$\delta_{ir} = \Phi_z \xi_{ir} \tag{B.1}$$

$$\hat{\xi} = \Phi_z^+ \Phi_z \xi_{ir} = \Phi_z^+ \delta_{ir} \tag{B.2}$$

Now the since only Φ_z is tracked using the MRS, the effective mean reconstruction error δ_{eff} can be expressed as:

$$\delta_{eff} = \frac{\| \begin{pmatrix} 0_{n\times 10} & 0_{n\times 10} & \Phi_z \end{pmatrix}^T \hat{\xi} \|_2}{n} \tag{B.3}$$

For each random sampled δ_{ir},we generated $\hat{\xi}$ and calculated δ_{eff}.
The associated δ_{eff}, $\delta_{ir} \sim \mathcal{N}(0, \sigma^2)$ average results over 100000 samples is summarized in table B.1:

Table B.1: Modal Noise Summary

$\sigma(mm)$	$\delta_{ir}(mm)$	$\delta_{eff}(mm)$	$\frac{\delta_{eff}}{\delta_{ir}}$
0.15	0.1192	$7.944 \cdot 10^{-4}$	0.666 %
1.5	1.192	$7.927 \cdot 10^{-3}$	0.665 %
15	11.92	$7.942 \cdot 10^{-2}$	0.666 %
150	119.2	$7.951 \cdot 10^{-1}$	0.667 %

As we can see in table B.1, the projection of the measurement error onto the modal space of the first 10 modal

shapes is consistently around 0.666 % of the MRS error. Note that usually, the error vector is not perpendicular to the z-axis, thus δ_{eff} is actually an upper bound on the error. Since the transformation between the modal displacement space and the spanned space is linear, this error represents the measurement error for every deformation of the wing. Figure B.1 samples of deformations generated with noise from distributions similar to the MRS noise distribution. As we can see, even when a large amount of noise is introduced, the projection of the displacement onto the modal-space dissipates nearly all the noise. The cause of this phenomenon is unclear and requires further study, however this significant discrepancy means the 100% metric should not be compared to the MRS error, but rather to its projection onto the space spanned by the natural modal shapes.

Figure B.1: Visualizations of δ_{eff} and δ_{ir} for different δ_{ir} distributions