Deep Learning project

Group 15

Group members:

Bjarke Holm Jørgensen, Jonathan Boel Nielsen, Kasper Møller Hansen, Thomas Seeberg Christiansen

Presentation of task

This project aims to determine the precise state of a wind turbine before initiating autonomous drone inspection. When a wind turbine is brought to a stop, it halts in an arbitrary configuration due to the lack of precise control over the positioning of its moving parts during shutdown. This random state introduces challenges for autonomous drones tasked with performing visual inspections, as they require detailed knowledge of the turbine's configuration to plan an effective scanning trajectory.

Specifically, the halted state includes the orientation of the turbine blades and the rotational position of the nacelle (turbine housing), which are necessary knowledge for planning an optimized flight path that the drone can follow around the structure.

Presentation of dataset(s)

For this project the dataset is generated from scratch using NVIDIA Isaac Sim, a 3D simulation environment. The 3D environment created was a flat and sparse scene having a ground plane and a skybox with varying textures for more robust training and a single 3D model of a wind-turbine. The simulation environment creates random poses for the wind turbine, controlling the base rotation, meaning the direction in which the wind-turbine nacelle is pointing, and the blade rotation. The base angle is randomized between 0° and 360°. The blade angle is randomized between 0° and 120° because of the blades' rotational symmetry. For each pair of angles two images are generated from two cameras which are the same height and distance from the wind turbine, but the second image is moved 90° counterclockwise along the radius from the windmill as can be seen in figure 1 and figure 2.

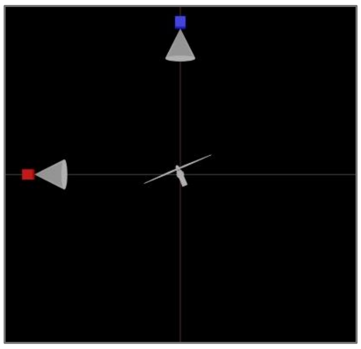

Figure 1. The blue box and red box represent camera_1 and camera_2 respectively. Both capture the wind turbine placed at the center of the scene.

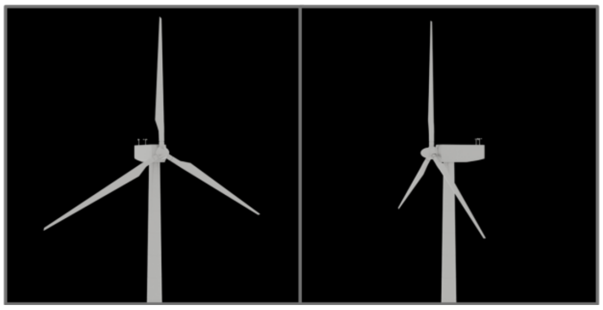

Figure 2. The left and right image pair correspond to the view from camera_1 and camera_2 respectively.

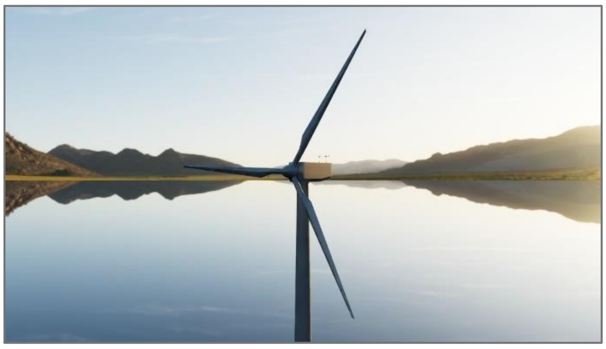

Figure 3. Sample image with background and ground plane.

The distribution of angles is uniformly distributed in their angle space. The base angle related to the image pair are all relative to camera_1, so that when the base angle is zero the nacelle points straight towards camera_1. The blade angle is zero when one blade is pointing straight up. Increasing the angles means rotating the base or blades counterclockwise.

9000 data points are generated consisting of a base angle, a blade angle and two images corresponding to the wind turbine rotating to those angles. Since the base angle rotation relative to camera_2 is simply 90 degrees less than for camera_1, it is possible to instead consider the image pair as two unrelated images where the image for camera_2 simply has the base angle subtracted by 90° and then normalized within 0° and 360°. Thus, we can also use only one image per angle pair, and then have 18000 data points. We will use the configuration with 18000 data points for our training, since we got better performance. Additionally, we train on data points generated with rotation angle intervals of $\pm 60°$, $\pm 45°$ and $\pm 30°$. This will be further elaborated on in the training strategy section.

Presentation of network structure

A Residual Network (ResNet) was selected for this project as it has exceptional performance for feature extraction and classification in computer vision datasets. Such a network is designed to address the vanishing gradient problem by employing skip connections that bypass one, or more, layers. This approach allows the construction of deeper networks without impacting the performance.

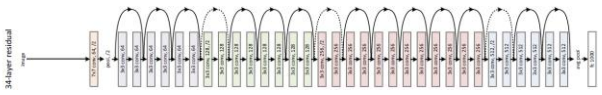

Figure 4. Original ResNet-34 structure.

For this project, we chose ResNet-34 as it strikes an effective balance between network depth and computational efficiency. This network was constructed with 34 layers and used for training on the ImageNet dataset to provide general-purpose image classification across 1000 categories. The original output layer had these 1000 output features, but since we have a regression problem trying to estimate two angles, we only need two output features: Blade angle and nacelle rotational position. Hence, we replaced the fully connected linear layer with two output features.

Although pretrained models offer valuable feature extraction capabilities, we opted to train the network from scratch. This decision was made because the specific features of wind turbines, particularly the blades and nacelle, may not be adequately represented in models pretrained on ImageNet. Training from scratch with our dataset ensures that the model learns features specific to wind turbines.

The images in our dataset are generated with a resolution of 1280x720 pixels, capturing a large area besides the wind turbine. Using these images would not be efficient since we only need to see the nacelle and blades. Hence, we cropped the images to a resolution of 350x350 centering on the nacelle. These resized images are then passed through the ResNet-34 network. The network is constructed with several convolutional layers of varying sizes. The initial layer has 7 input channels, 64 output channels, a kernel size of 7, and stride 2. This is followed by a 3x3 max pooling layer with stride 2. The remaining convolutional layers are of fixed size 3x3 but with an increasing number of filters from 128 to 512, while every second layer has a skip connection. Finally, an average pooling layer is used before the fully connected linear output layer.

Presentation of training strategy

We train two different strategies to compare them to see how well they perform. Method 1 is the practically easier method but will likely perform poorer than method 2, which is more complex. Method 1 involves using 18000 datapoints to estimate both the base angle and the blade angle from one image. Method 2 involves first using the 18000 data-points to estimate the base angle from one image, simulating an initial guess to where a drone should fly to place itself in front of the nacelle of the windmill. Then a second network is trained on images where the base angle rotation is limited to within $\pm 60°$, $\pm 45°$ and $\pm 30°$ degrees relative to the camera. The second network then estimates both base angle and blade angle from this dataset and should be able to do so more accurately because the relative rotation is easier to estimate when the nacelle is pointing towards the camera.

We preprocess the images before feeding them into our neural network. The images are cropped and downscaled to 350x350 pixels centered around the nacelle. To augment the data, we added an x-y offset in the cropping. These offsets are generated from a zero-mean normal-distribution with a standard division of 15. To get a zoom-variety, we cropped between 400x400 to 600x600 and down scaled to 350x350. By implementing the data augmentation in the transformation, the network would never see the exact same image twice, which would help with the problem of overfitting the model and training a more robust model.

For the training process we use the ADAM optimizer for faster convergence, and we use a variable step size using the PyTorch scheduler ReduceLROnPlateau(), which reduces the learning rate by a factor after a set number of epochs where the loss has not fallen. This will automatically adjust the learning rate every time we reach a plateau, which will help us approach our optimal state.

To accommodate our problem of estimating two different angles, each with a corresponding periodicity, P, we opted to make our own loss function. This function should, e.g. for the nacelle, yield no loss for a prediction of 360 degrees with the target being 0 degrees. Similarly, for the blades we needed to include a periodicity of 120 degrees. For each loss contribution the absolute error is shifted by half the period of the corresponding angle to ensure that angle differences are mapped be-

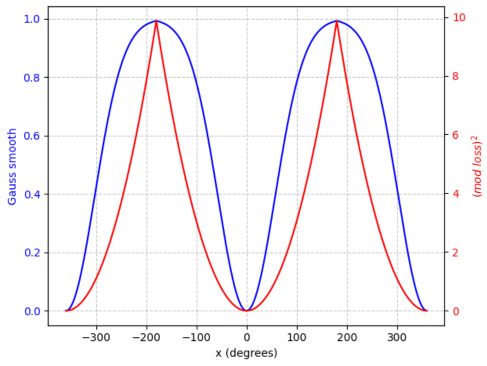

Figure 5. Modulus loss function for base angle.

tween zero and half the period. Taking the modulo with the period returns the remainder of division from which subtracting half the period yields the correct angle deviation.

$$err_i = |pred_i - target_i|$$

$$err_{mod,i} = \left(err_i + \frac{P}{2}\right)(mod\ P) - \frac{P}{2}$$

To normalize the loss and obtain a smooth loss function characteristic, the modulo-error is passed through a Gaussian-like smoothing function with $\sigma = 1$.

$$G(x) = 1 - \exp\left(-\frac{x^2}{2 \cdot \sigma^2}\right)$$

For each batch of size N, we compute the mean of each loss component and sum them for the total loss.

$$Loss = \frac{1}{N}\sum_{i=1}^{N} G(err_{mod,i})$$

$$Loss_{total} = Loss_{nacell} + Loss_{blades}$$

With this loss function errors in either the nacelle prediction or blade predictions are weighted equally in the combined loss.

The network is trained for 40 epochs after which the training does not improve further. The model captures the states of the wind turbine well and obtains more than 95% accuracy within 10 degrees in both base and blade angle.