# Deep counting of fir cones on larch trees

Matthieu Paques

18/11/2021

## Introduction

Tree development is conditioned by a trade-off between three main features: reproduction, growth and survival. Whilst the growth and the survival are easily measured (respectively the diameter and height of the trunk and whether the tree survived or not), the reproductive ability of the tree is a more demanding task as it requires to count the number of fruits produced. In the following work we will focus on counting the fruits of larch trees i.e. fir cones using Machine Learning techniques of computer vision. We wish to develop a model able to predict the number of cones on a particular tree using a RGB picture as an input.

Fir cones of larch trees are ovoid, brown and can reach about 4cm long and 3cm in diameter. Interestingly fir cones take two to three years to grow and fall implying a large diversity of sizes at any period of the year on a same tree. The counting of fir cones is harden by their dense distribution on branches. Their brownish colour also makes it hard to disentangle cones from the branches and the background. Finally, the task implies to count 3D distributed objects on a 2D picture. Fir cones behind the tree might be missed.

## Methods

### State of the Art

Two categories of models are used to realise object counting: counting by detection and counting by regression. Counting by detection relies on an explicit visual detection of every individual object instances in the image.The number of instance gives the desired output (Sermanet et al. 2014). Counting by regression avoid the explicit recognition of the objects. Instead, the output gives directly the desired number in an end-to-end approach (Oñoro & Lopez-Sastre 2016). The detection of the objects is realised implicitly by the model. On the one hand, counting by detection requires a more complex labelling of the training data (bounding boxes, segmented objects). On the other hand, counting by regression required a much bigger volume of labelled data whilst the labelling is simpler (expected number of objects). Multiple works have been made on counting fruits like tomatoes (Rahnemoonfar & Sheppard 2017), oranges (Liu et al. 2017), peers and apples (Itakura et al. 2021). One could note that most of aforementioned work concerned colourful fruits easily distinguishable from their environment. Little effort has been made on counting pine cones. As far as our knowledge, a single paper presented a machine learning solution for pine cones counting (Luo et al. 2020). Luo et al. presented an interesting combination of a BEGAN (artificial image generator) and a YOLOv3 (object detector) working in pairs. However, the model was applied to an easier-to-detect situation. On the pictures used, cones were lying on the floor instead of being hanged on a tree and only a few instances were shown compared to several hundreds in our case.

## Available Data

We possess 51 RGB pictures (size 1920*1440*3) of larches trees and their environment. On each picture the tree to consider is centred. For each tree, two measures are available: an "exhaustive" counting and a "manual" counting. The first data refers to a counting performed on site by technicians. The second measure was obtained by manually counting the cones on a picture using the ImageJ software. This manual counting tried to respect the following rule: counting the fir cones of the branches in the foreground while ignoring the rest. A plot of the manual counting versus the exhaustive counting shown a strong linear relationship between the two measures ($r^2 > 0.9$).

## Deep counting

Counting the fir cones of a particular tree requires first to extract the considered tree from its neighbours. We want to implement a first deep model having a picture as an input and the bounding box of the central tree as an output. Then, a second deep model will extract every cones from the picture. The final results will be obtained by counting the cones inside the bounding box.

The "tree model" in charge of extracting the central tree has several candidates. A first attempt was realised using the object detector network YOLO v3 (Redmon et al. 2016). Bounding boxes training labels were made on the online tool Roboflow (*Roboflow: annotation website* n.d.). Data Augmentation increased the training dataset up to 84 images. However, the training revealed poor results with predicted bounding boxes hardly disentangling one tree from another. The small size of the dataset and the known limited ability of YOLO to detect densely packed objects can reasonably explain this failure. A second attempt relied on semantic segmentation of the trees using a UNet network (Ronneberger et al. 2015). The output was a binary mask highlighting the trunk and the main branches of the different trees. The boundaries of the trees were obtained by considering the connected components of the mask (i.e. area sharing the same pixel intensity values). This second approach appeared to be more promising.

Similarly, the "cones model" realising fir cones extraction was decided to be a segmentation model. Considering the small size of the object (a few dozens of pixels) and the large number of instances per images (a range between dozens and thousands) the choice of a classic object detector model as YOLO or RetinaNet was ruled out. Instead, we preferred a more implicit approach relying on segmentation. The model had as an input the image of the trees and as an output a binary mask with the fir cones extracted. The UNet network and its improved version the UNet++ network (Zhou et al. 2018) were both tried. One could measure the coordinates and the number of cones by counting the number of local maxima of the mask. Finally, the number of cones of the central tree is obtained by discarding any points whose coordinates are outside the bounding box.

Considering our 2D input image of a 3D tree, one can expect a significant difference between the predicted number of cones and the exhaustive number. However, as for the manual counting and the exhaustive counting, we hoped to find a relationship between our deep counting and the exhaustive counting.

## Training

### Tree model

The tree model is a UNet network having an encoder with four down-sampling convolution layers and a decoder with four up-sampling layers. The image is reduced to dimension (832*832*3) to decrease the number of training parameters. The binary mask of the trees were made on

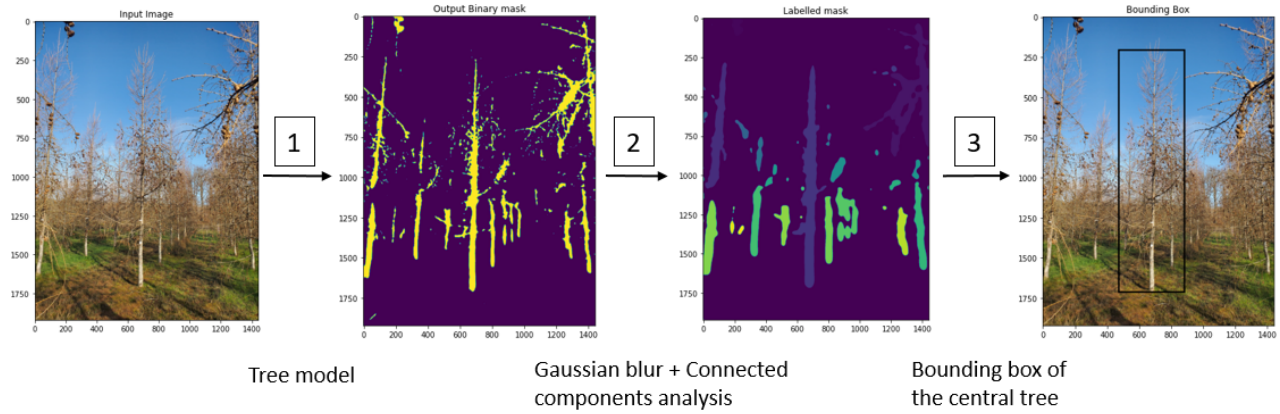


**Figure 1:** Extraction of the central tree

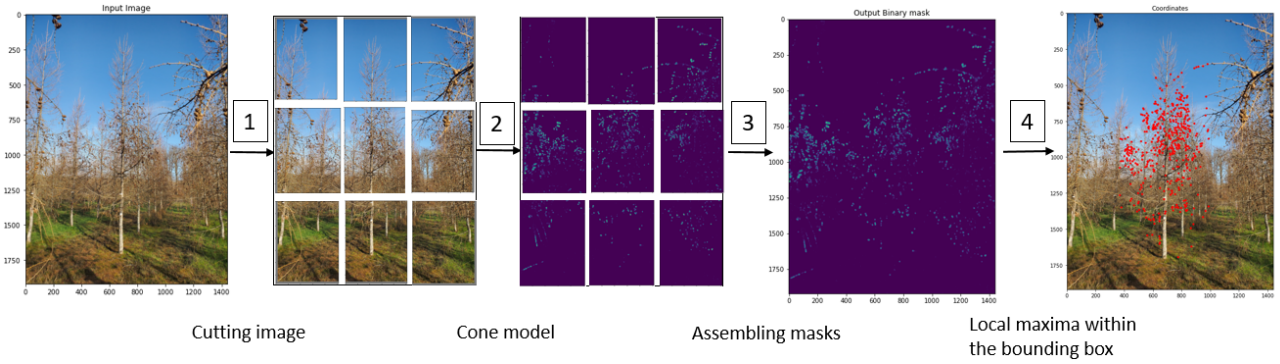


**Figure 2:** Extraction of the cones

the website (*Apeer : annotation website* n.d.) for 30 images. The model was trained using Data Augmentation (shear, rotation, zoom, width shift  height shift) and Early Stopping to avoid over-fitting.

### Cone model

The cone model was whether a UNet or a UNet++ model. In both cases, the input was a picture of the trees and the output was a binary mask noticing fir cones. Labelled binary mask were performed on the website (*Apeer : annotation website* n.d.) for 41 images. To increase the number of training data and reduce the number of network parameters, the images and masks were cut into sub-images. We tried to cut by 3 or 5 regarding the height and width with an overlap of 16 pixels. The division of the images resulted in respectively 9 sub-images of size (512*672*3) or 25 sub-images of size (320*416*3). The final predicted mask is obtained by assembling the predicted sub-masks reduced from the overlap margin. The overlap was added after the appearance of perturbed behaviour at the edges of the predicted sub-mask. The models were trained using Data Augmentation and Early Stopping to avoid over-fitting.

## Results

The performances of the tree model and the cone model are shown in table  1. We chose the Dice coefficient as a metrics to assess the similarity between the target mask and the predicted mask. The loss was set as the binary-cross entropy. We investigated what protocol gave the best performances for the cone model. One can see the smallest loss was obtained for the UNet++ model with an input image divided by 3*3. This settings were chosen for the following of the work.

| Cone model performances | | |
|---|---|---|
| $n_{cut}$ | UNet | UNet++ |
| 1 | loss=0.0341 - dice=0.2117 | loss=0.0349 - dice=0.1553 |
| 3 | loss=0.0351 - dice=0.2468 | loss=0.0296 - dice=0.2001 |
| 5 | loss=0.0381 - dice=0.2428 | loss=0.0309 - dice=0.1619 |
| **Tree model performances** | | |
| 1 | loss=0.0764 - dice=0.4148 | X |

**Table 1:** UNet (1,941,105 trainable parameters) was trained with a batchsize of 16 and 30 epochs with 100 steps per epochs. UNet++(2,049,681 trainable parameters) was trained with a batchsize of 8 and 30 epochs with 100 steps per epochs. Difference of batchsize was imposed by a lack of GPU memory (Google Collaboratory GPU P100).

We assessed the performance of our technique by plotting the exhaustive counting versus the deep counting (see figure 1). We also plotted the exhaustive counting versus the manual counting as a comparison. A power transformation was needed to improve the linear relationship between the 2D counting and exhaustive counting ($log(\hat{y}) = \beta 0 + \beta 1 * log(x)$). Our deep counting shown promising results exerting a strong correlation coefficient ($r^2 = 0.80$) while being still under the manual counting method ($r^2 = 0.94$). A quick look at the residuals plot shown the linear regression model was a relevant choice for the deep counting, with the residuals being normally distributed around 0. The appearance of a decreasing pattern for residuals of the manual counting indicates a linear regression might not be the best The plot of the residuals also allowed to detect the main outliers. On a normalized residuals plot, we consider as outliers points outside $\pm\sqrt{3}$. In our particular case, removing the 3 worst outliers improved the fitting of the curve with a new Pearson's coefficient $r^2 = 0.89$. The best and worst predictions are shown in Appendix Figure 5 and Figure 6. It appeared that our model performed well on images where the central tree is clearly separated from the background. Without any surprise, best predictions were also obtained for clearer images. Codes can be retrieved from (Paques 2021).

# Discussion

A main criticism that can be done on our work is the choice of a bottom-up approach. Extracting the trees, finding the bounding box of the central one, extracting the cones and counting them are four actions performed independently in our strategy. We could instead realise the same work in an end-to-end approach. The first step would be the same: we train a model to segment the fir cones (segmentor). Then, we build another model as the superposition of the segmentor whose weights are frozen and a regression model. The regression model would be composed of multiple dense layers with the final layer having a single output and the loss being the mean squared error. Such a model would be able to encode all the aforementioned tasks in an optimised way. However, it might require more training data to be effectively trained.

A pre-processing attempt was performed on the image using the Sobel filter. A Sobel filter extract the edges of an image by computing the gradient of the pixel intensities. Despite being commonly employed for object recognition it didn't give probing results in our case. A straightforward explanation could be the prepossessed image is grey-scaled and colour may be a determinant feature in tree and fir cone detection. Other image transformations were tried like HUV (Hue, Saturation, Value) and YUV (luma component, blue projection, red projection) without any success. An appealing method to emphasise the fir cones from the environment could be the use of infra-red images instead of RGB images (see Figure 4). Indeed, fir cones have the ability to store heat during the day. Therefore they present a slightly higher temperature than the branches and the trunk what ease their detection (Takács et al. 2008).

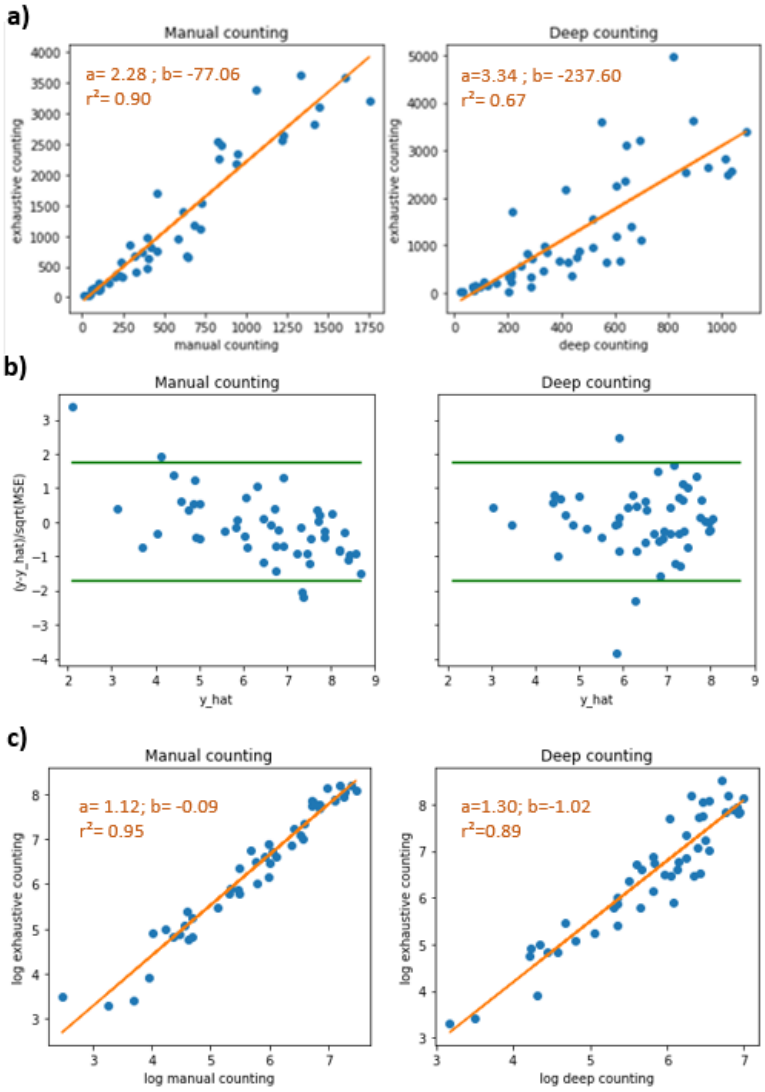


**Figure 3:** Left: manual vs exhaustive counting. Right: deep vs exhaustive counting. **a)** Linear Regression. **b)** Normalised residuals after power transformation. Green lines corresponds to $f(x) = \pm\sqrt{3}$. **c)** Linear Regression after power transformation and removing of the three worst outliers.

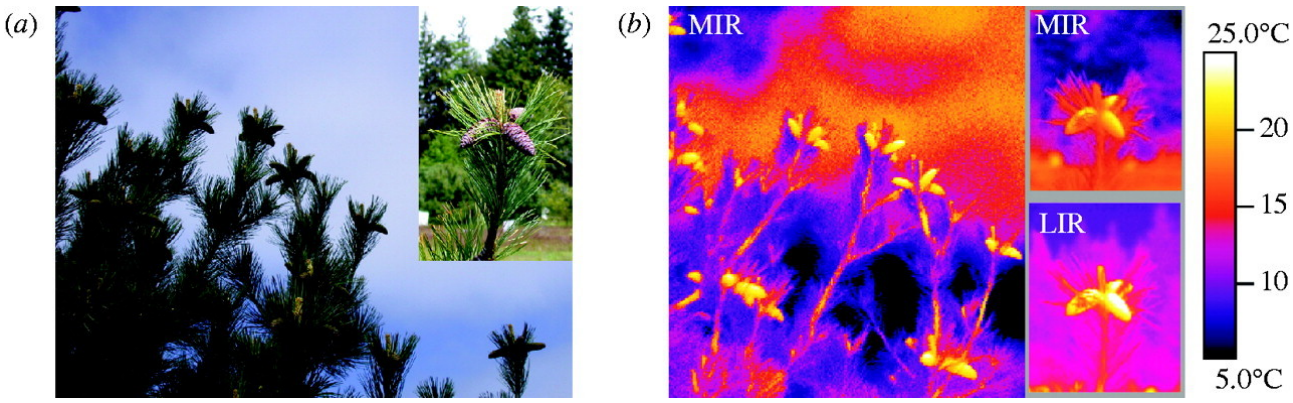


**Figure 4:** Infrared picture of western white pine cones in mid-range (3–5m; MIR) and long-range (8–20m; LIR) IR spectrum thermographs. Retrieved from (Takács et al. 2008).

For further development of machine learning models several suggestions can be made regarding the data collection. A Machine Learning model works well on type of data it has already seen during training. To make the model invariant to weather hazards we would suggest to take pictures under the most diverse conditions (amount of sun, rain, time of the day, season). We also recommend using a high resolution camera (image size > (1920*1440*3)) and paying attention to realise a good focus (clear foreground, blurred background). Furthermore, a simple trick to have more data would be to take multiple pictures of the same tree. It would increase the training dataset while avoiding to count more trees exhaustively. We would like to warn against a possible bias regarding the cardinal direction of the photo shot. Indeed, it seems that for the Canadian Red Pine, the distribution isn't uniform across the four cardinal points with a significant increase of the number of cones for branches oriented to the South (Mattson 1979). Larch trees might experienced the same kind of cardinal variance regarding the cones distribution.

# Conclusion

The objective of this work was to investigate the possibility of a machine learning solution to realise fir cones counting on larch trees. We opted of a bottom-up approach realising separately the extraction of the central tree, the extraction of the cones and finally the counting. We obtained promising results with predictions being strongly correlated to their target values ($r^2 = 0.80$). We discussed the opportunity to realise a more effective model by considering an end-to-end approach. Finally, we made several suggestion regarding the future collection of data. Efforts should be made on increasing the variability of light conditions while increasing the repeatability of the photoshot (same cardinal orientation, sharpness of the image, focus).

# References


*Apeer : annotation website* (n.d.), https://www.apeer.com/app/machine-learning/annotate.

Itakura, K., Narita, Y., Noaki, S. & Hosoi, F. (2021), 'Automatic pear and apple detection by videos using deep learning and a kalman filter.', *OSA Continuum* .

Liu, X., Chen, S. W., Aditya, S., Sivakumar, N., Dcunha, S., Qu, C., Taylor, C. J., Das, J. & Kumar, V. (2017), 'Robust fruit counting: Combining deep learning, tracking, and structure from motion'.

Luo, Z., Yu, H. & Zhang, Y. (2020), 'Pine cone detection using boundary equilibrium generative adversarial networks and improved yolov3 model.', *Sensors* .

Mattson, W. J. (1979), 'Red pine cones: distribution within trees and methods for sampling', *Canadian Journal of Forest Research* **9**(2), 257–262.

Oñoro, D. & Lopez-Sastre, R. (2016), 'Towards perspective-free object counting with deep learning', *In International Conference on Medical image computing and computer-assisted intervention* .

Paques, M. (2021), 'cone detector repository', https://github.com/Makeumitchel/cone$_d$*etector*.

Rahnemoonfar, M. & Sheppard, C. (2017), 'Deep count: Fruit counting based on deep simulated learning.'.

Redmon, J., Divvala, S., Girshick, R. & Farhadi, A. (2016), 'You only look once: Unified, real-time object detection', *In Proceedings of the IEEE conference on computer vision and pattern recognition* pp. 779–788.

*Roboflow: annotation website* (n.d.), https://roboflow.com/.

Ronneberger, O., Fischer, P. & Brox, T. (2015), 'U-net: Convolutional networks for biomedical image segmentation', *In International Conference on Medical image computing and computer-assisted intervention* pp. 234–241.

Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R. & LeCun, Y. (2014), 'Overfeat: Integrated recognition, localization and detection using convolutional networks', *In International Conference on Medical image computing and computer-assisted intervention* .

Takács, S., Bottomley, H., Andreller, I., Zaradnik, T., Schwarz, J., Bennett, R., Strong, W. & Gries, G. (2008), 'Infrared radiation from hot cones on cool conifers attracts seed-feeding insects', *Proceedings. Biological sciences / The Royal Society* **276**, 649–55.

Zhou, Z., Siddiquee, M. M. R., Tajbakhsh, N. & Liang, J. (2018), 'Unet++: A nested u-net architecture for medical image segmentation', pp. 234–241.

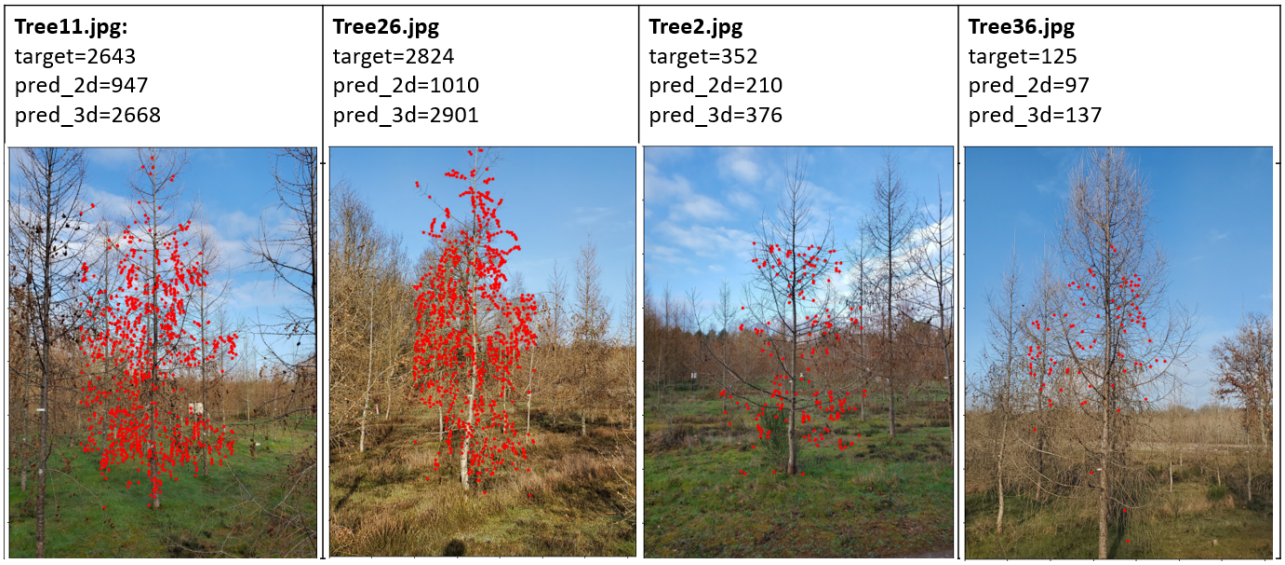


**Figure 5: Four best predictions** From left to right: smaller to bigger squared error between target and pred 3d.

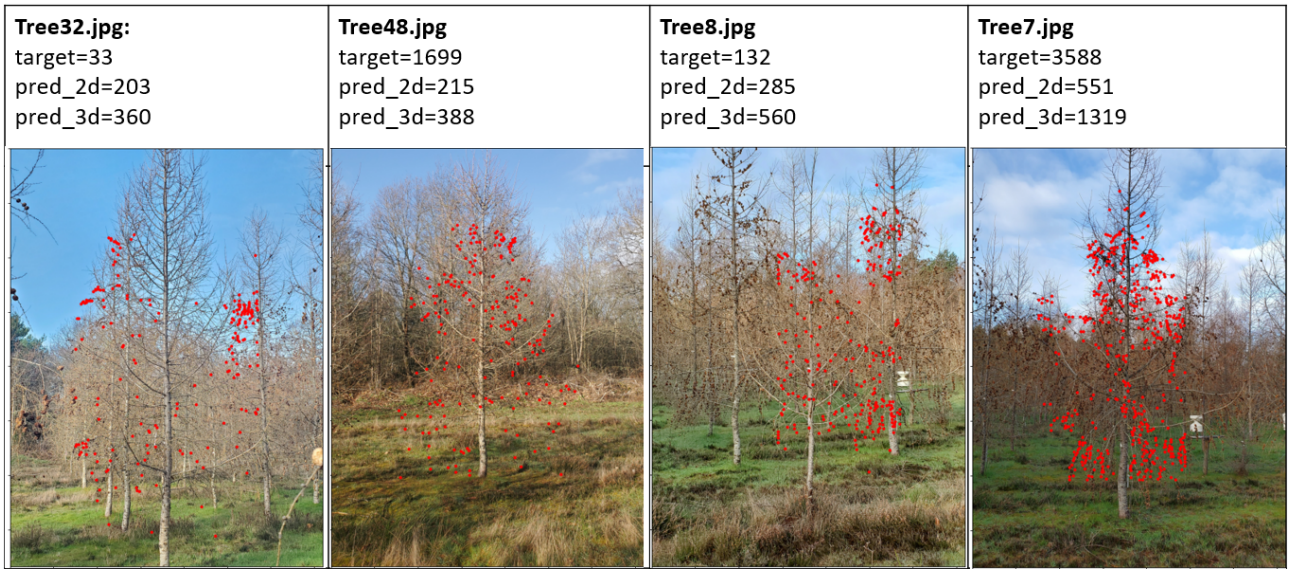


**Figure 6: Four worst predictions** From left to right: bigger to smaller squared error between target and pred 3d.

# Appendix