A kNN approach based on ICP metrics for 3D scans matching: an application to the sawing process

Sylvain Chabanet, Philippe Thomas, Hind Bril El-Haouzi, Michael Morin, Jonathan Gaudreault

Proceedings of the 17th IFAC Symposium on
Information Control Problems in Manufacturing
Budapest, Hungary, June 7-9, 2021

A kNN approach based on ICP metrics for 3D scans matching: an application to the sawing process ⋆

Sylvain Chabanet * **Philippe Thomas** * **Hind Bril El-Haouzi** *
Michael Morin **,*** **Jonathan Gaudreault** **,****

* *Université de Lorraine, CNRS, CRAN, F-88000 Epinal, France (e-mail: {sylvain.chabanet, hind.el-haouzy, philippe.thomas}@univ-lorraine.fr)*
** *FORAC Research Consortium, Université Laval, Québec, QC, Canada.*
*** *Department of Operations and Decision Systems, Université Laval, Québec, QC, Canada*
**** *Department of Computer Science and Software Engineering Université Laval, Québec, QC, Canada*

Abstract: The Canadian wood industry use sawing simulators to digitally break a log into a basket of lumbers. However, those simulators tend to be computationally intensive. In some cases, this renders them impractical as decision support tools. Such a use case is the problem of dispatching large volume of wood to several sawmills in order to maximise total yield in dollars. Fast machine learning metamodels were recently proposed to address this issue. However, the approach needs a feature extraction step which could result in a loss of information. Conversely, it was proposed to directly make use of the raw information, available in the 3D scans of the logs typically used by a recent sawmill simulator, in order to retain that information. Here, we improve upon that method by reducing the computational cost incidental with the processing of those raw scans.

Keywords: Sawing Simulation, Iterative Closest Point, Nearest Neighbours, Machine Learning Application

1. INTRODUCTION

As can be seen from the various literature on applications of artificial intelligence to the wood industry, machine learning could advantageously intervene at various levels of the forest-wood supply chain. For example, Raczko and Zagajewski (2017) compare several machine learning algorithms to map tree species using airborne data. Such tool would limit the need for more time consuming and labour intensive field observations. Nguyen (2018) exposes a method to detect and classify defects on tree trunks using terrestrial Lidar scans. Their goal is to assess the quality of a standing tree very early in the production process. Morin et al. (2020) propose to use machine learning metamodels instead of a slower sawing simulator to tackle the optimisation problem of wood allocation between several sawmills. Nasir (2020) study the possibility of using artificial intelligence approaches to monitor the cutting power and waviness during the sawing of a log. The benefit of this study is a better understanding of the sawing process, which, according to the author, could be used to implement decision making tools and improve the sawmill log recovery.

⋆ The authors gratefully acknowledge the financial support of the ANR-20-THIA-0010-01 Projet LOR-AI (lorraine intellgence artificielle) and région Grand EST. We are also extremely grateful to FPInnovation who gathered and processed the dataset we are working with.

In Thomas (2017), an artificial neural network is used to automatically grade sawed lumbers according to the National Hardwood Lumber Association rules. Such a neural network would need less computational power than other existing software implementing the full set of rules.

In this paper, we focus on the problem of predicting the collection of lumbers (we call a basket of products) obtained from the sawing of a log. This problem is rendered particularly difficult by the divergent nature of the sawing process, i.e., when breaking a log into lumbers, several outputs, with potentially different characteristics, are produced at the same time in co-production (as depicted in Figure 1). Currently, there exist simulators able to

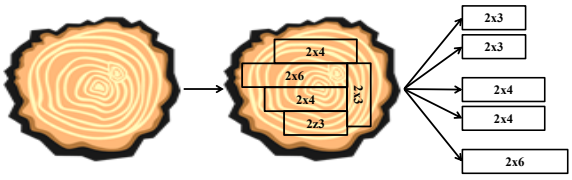

Fig. 1. The sawing of a log into lumbers is a diverging process with co-production. (Morin et al., 2015)

process 3D log scans in order to compute the basket

of products that would be obtained from the sawing of said log by a given sawmill. Using those simulators is, however, computationally intensive for short to medium term decision making. The allocation of large volume of wood to several sawmills is an example of such a problem (Morneau-Pereira et al., 2014; Morin et al., 2015, 2020). In order to maximise yield, several allocation scenarios need to be evaluated, each involving some allocation of the logs to sawmills and an evaluation of the value of that allocation. To get the value of an allocation, the value of each pair of log and sawmill in that allocation needs to be known. This is where the simulator comes into play. That is, it is used to approximate the output of a sawmill given a log in order to compute the basket value.

Alternatives were proposed to replace the sawing simulator by a much faster machine learning metamodel. Among them, Selma et al. (2017) propose to use an Iterative Closest Point algorithm (ICP) to compute similarities between pairs of wood logs. Given a log of unknown basket, those similarities are then used inside a 1 Nearest Neighbour algorithm (1NN) to return the basket of products of the most similar log as an output. The ICP algorithm was chosen for its simplicity and flexibility. Just as a sawing simulator, it works directly on the 3D scan of a log. In our data, a 3D log scan is a point cloud made of equally spaced ellipsoid sections where each section consists of points sampled randomly on the log surface. Although the number of points varies greatly between scans, the ICP is able to compute the similarity, hence its flexibility. However, the multiple runs of ICP needed to make a prediction render this solution computationally expensive (w.r.t. the use of a machine learning metamodel based on simpler characteristics such as dimensions and curvature). Hence, in this paper, we improve upon the ICP-based prediction approach by reducing its computational cost without lowering the quality of the predictions.

The paper is structured as follows. In Section 2, we overview the related literature. Section 3 presents our proposals to reduce the computational cost of a prediction. The experiments and the results are exposed in Section 4. We conclude in Section 5.

2. PREVIOUS WORKS

In this section, we summarise the literature relevant to the techniques we use to approximate the output of a sawing simulator. First, we review recent research on building sawing simulators metamodels. Then, we overview the ICP and the kNN algorithms concepts.

2.1 Sawing simulator metamodels

The Canadian wood industry has several simulators like SAWSIM, Optitek or Autosaw available to digitally simulate the breaking of a log into lumbers (Wery et al., 2018). Given the 3D scan of a log and the characteristics of the sawmill used, those simulators identify an appropriate sawing pattern to break the log into a basket of lumbers. However those simulations come at a non-negligible computational cost, especially when considering it should be ran for numerous logs, sawmills, and sawmill configurations. Morin et al. (2015) propose a method to solve this issue by

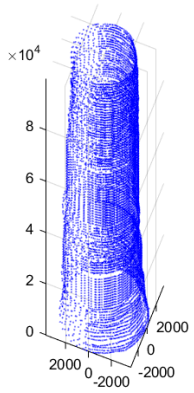

Fig. 2. The 3D scan of a log used as input for a simulator. Here, the points are not uniformly sampled on the log surface

replacing the simulator with a machine learning metamodel. Such a metamodel would be able, given some inputs describing a log, to yield a prediction for the output of a simulator with fixed parameters. Those metamodels would use only a fraction of the computational resources needed by the real simulator to approximate the lumbers given a log and sawmill pair. Considering that those predictions can't always be exact, it amounts to trade precision for running time. Several machine learning algorithms were then trained to predict the basket of a log, using only a vector containing six parameters describing each log. Morin et al. (2020) extend this work to other datasets from different sawmills and consider more particularly the problem of using machine learning-based metamodels to allocate different logs to different sawmills. They also add the wood species to the six parameters used in their first study.

However, considering that reducing the logs to only a small vector of characteristics may lead to an important loss of information, Selma et al. (2017) propose a strategy using unstructured data, i.e., the 3D scans of the logs which contain unordered points sampled randomly on the logs surfaces. One particularity of 3D log scans is that the number of points vary greatly from one scan to another rendering a direct use of machine learning, i.e., without features extractions or adapted approaches, difficult. Selma et al. (2017) proposes an adapted approach based on the ICP algorithm. In the paper, they use the algorithm to compute a similarity between pairs of logs and use it in a 1-nearest neighbour machine learning model (1NN). More specifically, given a new log whose basket needs to be predicted, this model first computes how similar the log is to all known logs. Then, it selects the most similar log, in terms of ICP distance, and attributes that basket to the unseen log. That method was underperforming at first due to missing sections in the 3D log scans. A simple interpolation that repeats the immediately preceding section resulted in an impressive increase in performance.

2.2 The Point-to-point ICP

Point-to-point ICP was first introduced in Besl and McKay (1992) as a solution for the problem of shape registration.

Consider two point clouds sampled from two regions on an object surface. Those sampled areas can either be the same area sampled twice, or different but partially overlapping areas. In the following, one of those point clouds will be called the *target* shape whereas the other will be called the *source* shape. The ICP algorithm aims at finding a rigid transformation to apply on the source so as to align the two shapes as if they came from a single sampling of the object surface. The steps of that algorithm are as follows.

(1) **Pairs selection**. Pair each point in the source shape with its closest counterpart in the target shape. There is N_s such pairs, noted (s_i, t_i) with s_i and t_i being three dimensional points in the shape and target respectively and N_s the number of points in the source.
(2) **Construction of a metric**. This algorithm considers the metric $E(R,T) = \frac{1}{N_s}\sum_{i=1}^{N_s}(Rs_i + T - t_i)$ where R and T are respectively the rotation and translation on which E has to be minimised. This metric measures a similarity between the transformed source shape and the target shape.
(3) **Metric optimisation**. Find a pair of rotation and translation that minimises E. Such a pair form a rigid transformation.
(4) **Transformation** Apply this transformation to the source, go back to point two and loop until some ending criterion is achieved. For example, Besl and McKay (1992) stop the iterations when the metric decrease between two iterations of the loop pass under a threshold.

Several methods exist to find a closed-form solution to step 3 minimisation problem. One of them rely on the quaternion theory. Consider a unit quaternion $q = (q_0, q_1, q_2, q_3)^T$ with $q_0 > 0$. This quaternion represents a rotation of angle θ around the axis $\overrightarrow{u}$ where $q_0 = cos(\frac{\theta}{2})$ and $(q_1, q_2, q_3)^T = sin(\frac{\theta}{2})\overrightarrow{u}$.

To find the optimal transformation we first need to compute the quaternion matrix. To this end, let $\overline{s}$ and $\overline{t}$ represent the centre of mass of the source and of the target respectively, i.e., $\overline{s} = \frac{1}{N_s}\sum_{i=1}^{N_s} s_i$ and $\overline{t} = \frac{1}{N_s}\sum_{i=1}^{N_s} t_i$. The cross-covariance matrix of the point clouds is then defined by

$$\Sigma_{ST} = \frac{1}{N_s}\sum_{i=1}^{N_s}(s_i - \overline{s})(t_i - \overline{t})^T . \quad (1)$$

Let Δ be such that $\Delta = (H_{23}, H_{31}, H_{12})^T$ where H_{ij} are the elements of the matrix $H = \Sigma_{ST} - \Sigma_{ST}^T$ and compute the 4×4 quaternion matrix as follows:

$$Q = \begin{pmatrix} tr(\Sigma_{ST}) & \Delta^T \\ \Delta & H - tr(\Sigma_{ST})I_3 \end{pmatrix} \quad (2)$$

where I_3 is the 3×3 identity matrix.

Given matrix Q, the quaternion which represents the optimal rotation R is then simply the eigenvector associated with the largest eigenvalue of Q.

The optimal translation T is the translation which aligns the mass centre of the source, after rotation, with the mass centre of the target. It can then be computed as $T = \overline{t} - R\overline{s}$.

The final value of the metric E computed at the end of the last loop of the ICP can be considered as a similarity between the source and target point clouds.

2.3 The k-nearest neighbours algorithm (kNN)

The kNN rule (Fix and Hodges, 1951) is a very well-understood algorithm commonly used in supervised learning. We recall that in our case, the supervised learning problem is to predict the basket of products of a new log, using a dataset of known logs and baskets. The kNN principle is as follows. Given a new data point (a wood log in our case), the rule considers its k nearest neighbours in the known dataset, i.e, the k points in the known dataset with minimal distance to the new point. Let's call $c_1, ..., c_k$ the labels of those neighbours. The set of all possible labels is noted C. The prediction yielded by the kNN algorithm for this new log is then

$$\hat{c} = \arg\max_{c \in C} \sum_{i=1}^{k} \frac{1}{k} I_{\{c_i = c\}} , \quad (3)$$

where

$$I_{\{c_i=c\}} = \begin{cases} 1 & \text{if } c_i = c \\ 0 & \text{otherwise} \end{cases} . \quad (4)$$

Simply speaking, the algorithm counts how often each label appears in the k neighbours and predicts the most frequent one. In some cases, two class c_1 and c_2 can obtain the same number of votes. In that case, one can search the closest neighbour among the neighbours having class c_1 or c_2 and choose its class as the winner.

3. PROPOSALS

We propose several changes to the ICP+1NN strategy exposed in Selma et al. (2017). Concerning the nearest neighbour classifier used after the ICP comparisons, we propose to replace the simple 1NN classifier by a kNN classifier. The hyperparameter k is chosen by cross-validation, as is common in the field. In addition, we also make two propositions in order to reduce the cost of running multiple ICP during the prediction process.

First, we propose to introduce a set of simple rules to decide when to run, or rather when not to run, an ICP to compute the similarity between two logs. Those rules are:

- Lumbers produced at a given sawmill have a lesser possible length. Hence, all new logs shorter than this minimum length can be declared having an empty basket of products without running any ICP. In addition, all such *very short* logs can be eliminated from the comparison database.
- Beside those *very short* logs and a few outliers, the logs come in a few standardised lengths. Based on that fact, it was decided to separate the remaining logs in the comparison database into groups of same length. When the basket of products of a new log has to be predicted, this log length will be computed, and the log will only be compared with other logs from the same length group in the comparison database.

Second, the most computationally expensive step of running an ICP is the pairing of the points between the target shape and the source shape. For example, the complexity of computing the nearest neighbours in the target shape of all the points in the source shape is $O(N_S \ln N_T)$ when using k-d trees (here, N_S is the number of points sampled from the source log, and N_T is the number of points sampled

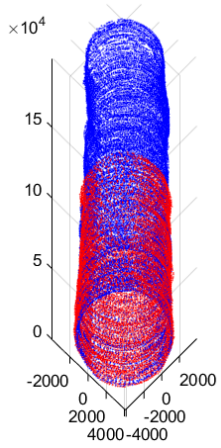

Fig. 3. Two logs having a small ICP distance despite being of different lengths. When fitting the red log on the blue one, the algorithm fits the red points with only the lower half of the blue points. Indeed, the ICP algorithm is designed to work with shapes overlapping only partially.

from the target log) (Besl and McKay, 1992). We propose to resample from those points to reduce their number.

The down-sampling method chosen was voxels down-sampling which proceeds as follows. The space containing the point cloud is first partitioned into squares of length L called voxels. Then, all the points in a voxel are replaced by their mean. The parameter L has to be chosen large enough to reduce considerably the number of points, but not too large to not downgrade the quality of the predictions. We tried several possible voxel length in preliminary experiments. It appears, from our experiments, that the value $L = 2.5$ cm work well. For comparison, the average distance between a point and its nearest neighbour in our 3D log scans is 0.3 cm. By choosing $L = 2.5$ cm, we reduce the number of points to about 16% of it's original value. Although, the total reduction varies a lot between logs. Another expected advantage is linked to the fact that the points in the scans are not uniformly distributed w.r.t. the original logs' surface. Some of the logs have points placed far more densely on one of their faces than on the other.

4. EXPERIMENTS

In the first three following subsections, we describe the data used for the experiments, the metric used to calculate the quality of a prediction, and the model building methodology. Finally, in the last subsection, we present the results obtained using the proposed ICP-based kNN variants.

4.1 Description of the database

All experiments were conducted using the dataset of Selma et al. (2017). This dataset contains 1,207 scans of logs and their related basket of products computed by the simulator Optitek (FPInnovations, 2014). A scan is a numeric table $N_{point} \times 3$, with N_{point} the number of points. Each column corresponds to one spacial coordinate. Each of those scans contains 12,000 points on average. The missing sections in the scans were filled by simply repeating the section of the log that precedes them.

Based on the lengths of the logs and lumbers present in our database, we separated the logs into three groups. The *very short* ones whose length is under the minimum length of a product, 247 cm, the *short* ones of length 254 cm and the *long* ones of length 503 cm.

Sawing a log into lumbers being a diverging process with co-production, the basket of products associated with a log is a list containing what lumbers would be obtained from sawing it. Each lumber is characterised by its dimensions (width, length, and depth), as well as its grade. The grade of a product is an ordering relation between lumbers sharing the same dimensions and represent its quality. In the Canadian wood industry, from which those data originate, the grade of a product is determined according to the National Lumber Grade Authority rules. Originally, the dataset had up to 47 different possible products. To simplify the prediction problem, the products were merged by grade so as to reduce their number to 19. For machine learning purposes, the baskets of products are represented by a sparse vector of length 19 where each entry corresponds to a product count. Due to the volume of the logs, baskets from this dataset contain at most five different kinds of products.

To evaluate and compare our proposed strategies, the dataset was randomly partitioned into two: A train set of size 724 and a test set of size 483.

4.2 The prediction/production score

In order to compare the efficiency of two machine learning models, one has to define a way to measure how precise the predictions they yield are. In this paper, we use the prediction/production scores ($s^{pre\times pro}$) introduced by Morin et al. (2015) specifically for this problem.

Given a log X, its corresponding basket Y, and $\hat{Y}$ the basket predicted for X by some metamodel, the $s^{pre\times pro}$ score is calculated in three steps, as follows.

First, we filter the vectors to avoid overly optimistic evaluations. We recall that Y and $\hat{Y}$ are sparse vectors of length 19. To eliminate the optimistic bias induced by the numerous $(0, 0)$ pairs in $(y_i, \hat{y}_i)_{1\leq i\leq 19}$, the vectors are filtered so as to eliminate those pairs.

Second the prediction and the production scores are calculated. The prediction score is defined by

$$s^{pre} = \frac{1}{p}\sum_{i=1}^{p} \min\left(1, \frac{max(\epsilon, y_i)}{max(\epsilon, \hat{y}_i)}\right) \tag{5}$$

with ϵ a small value (to avoid dividing by 0), and p the length of the filtered Y and $\hat{Y}$ vectors. This score can be seen as the average per product percentage of the prediction that is actually produced. In a similar way, the production score is defined by

$$s^{pro} = \frac{1}{p}\sum_{i=1}^{p} \min\left(1, \frac{max(\epsilon, \hat{y}_i)}{max(\epsilon, y_i)}\right) \tag{6}$$

and measures the average per product percentage of the production that is correctly predicted.

Finally, given the above definitions, the prediction /production score is calculated as

$$s^{pre\times pro} = s^{pre} \times s^{pro}\,. \tag{7}$$

The higher the prediction/production score, the better the model.

4.3 Methodology for Model Building and Evaluation

The models compared in this paper vary greatly in terms of runtime. Therefore, we do not want to compare the rules solely on the basis of their estimated risk. For that reason, the dataset was partitioned once and for all into a train set and a test set. For each model using a kNN classifier, a 10-Fold cross-validation was used on the train set to choose the value of the parameter k. The performance of the models were then compared on the test set.

Such a cross-validation procedure is commonly used in supervised learning applications to select the best hyperparameter for a machine learning algorithm. First, the train set S is partitioned in K subset of equal size, e.g., for $K = 10$, we have partitions $S_1, ..., S_{10}$. Then, for each value m of the hyperparameter one wants to compare, the cross-validation score of the associate model f_m using this hyperparameter value is computed as

$$CV_m = \frac{1}{10}\sum_{v=1}^{10}\frac{1}{|S_i|}\sum_{x,y\in S_i} s^{pre\times pro}(f_m(S_i^c, x), y)\,, \qquad (8)$$

where $\frac{1}{|S_i|}\sum_{x,y\in S_i} s^{pre\times pro}(f_m(S_i^c, x), y)$ is the average prediction/production score on S_i when using the model f_m as trained on S_i^c, i.e., on all the partitions except S_i. The cross-validation paradigm then dictate to choose the hyperparameter m maximising CV_m.

Once the hyperparameter of each model have been chosen by cross validation, their average prediction/production scores are evaluated on the test set and compared using a centred Wilcoxon signed rank test. Considering two models $f_1(D_{N_e})$ and $f_2(D_{N_e})$ trained on our train set D_{N_e}, the Wilcoxon test will consider the null Hypothesis.

$\mathscr{H}_0$: The median of the distribution of
$s^{pre\times pro}[f_2(D_{N_e}, X), Y] - s^{pre\times pro}[(f_1(D_{N_e}, X), Y]$
is equal to 0.

We are interested in negating this null hypothesis. In particular, having the median different from 0 would imply that one of the trained models will yield a prediction with a higher prediction/production score than the other with probability above 0.5.

It has to be stressed out that this method does not take into account the variability over the training set and doesn't allow for extrapolations on other training sets.

4.4 Results and discussion

In this section, the performances of six models are compared. Those six models are:

- **Model 1** : A random forest classifier trained following Morin et al. (2015) methodology.
- **Model 2** : The logs are classified according to their length, then only compared to logs of the same length in the comparison database. This model use a kNN classifier, with k chosen by cross validation.
- **Model 3** : The logs are classified by length and their scans filtered so as to reduce the number of points in each scan. This model uses a kNN classifier, with k chosen by cross validation.
- **Model 4** : Each new log in the test database is compared with all the logs in the comparison database. No filtering of the scans is done. This model uses a 1NN classifier.
- **Model 5** : The scans of each new log are filtered using voxel down-sampling. Each log is then compared with all the logs in the comparison database. This model uses a 1NN classifier.
- **Model 6** : A basket of products is assigned at random to each log in the test set. Those baskets are sampled from the set of baskets in the train database.

The simulations were run on an Intel Core i7 vPRO 10^{th} geneneration CPU at 2.70GHz.

For both models using kNN, i.e. Models 2 and 3, the cross-validation procedure selected the number of neighbours $k = 25$.

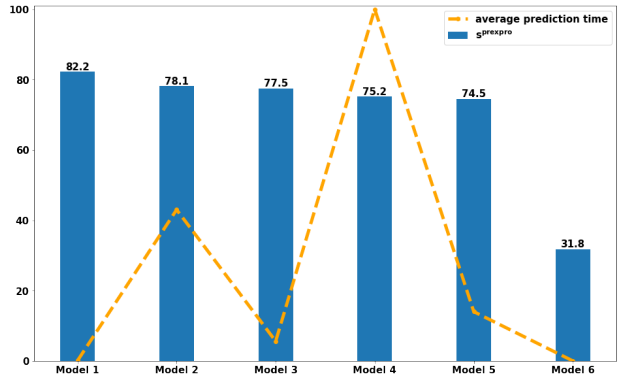

Fig. 4. Prediction/production scores for each model averaged on the test set. The figure also shows the average time needed to yield a new prediction, as a ratio of the time needed by Model 4, which is 21s.

Figure 4 displays the prediction/production scores of each model, averaged on the test set, as well as the average time needed to yield a prediction for a new log. Those times are displayed as ratios of the time needed by the simple ICP+1NN model (i.e., Model 4) to yield a prediction. This model needs in average 21 seconds to predict a basket of products for an unknown log. Model 2, has the higher score among the models using ICP. In particular, it is higher than Model 4. That increase in score comes from several factors. First, the number of neighbours used in the kNN classifier of Model 2 was fixed using cross validation and is therefore tailored for our specific problem. On the contrary, the number of neighbours used in Model 4 is arbitrarily set to 1. However, the increase in score due to the kNN replacing a 1NN is low compared with what is due to the classification of the logs by length. Indeed, further experiments show that using a 1NN instead of a kNN in Model 2 lower the score by less than 1%. However, due to the classification of the logs by length, all the logs in the *very short* class are now well classified. Similarly, due to the fact that the ICP algorithm is tailored to work on point clouds that are only partially overlapping, it sometime consider a *short* log to be very similar to a *long* one. Sorting the logs by length to only compare them inside a same length category solve that issue.

Table 1. P-values of the Wilcoxon tests. p-values under the 10% threshold are set in bold.

	Model 2	**Model 3**	**Model 4**	**Model 5**	**Model 6**
Model 1	$\mathbf{9.1 \times 10^{-4}}$	$\mathbf{1.8 \times 10^{-4}}$	$\mathbf{2.3 \times 10^{-6}}$	$\mathbf{2.0 \times 10^{-6}}$	$\mathbf{4.1 \times 10^{-58}}$
Model 2	-	1.2×10^{-1}	$\mathbf{1.0 \times 10^{-2}}$	$\mathbf{3.6 \times 10^{-3}}$	$\mathbf{2.4 \times 10^{-52}}$
Model 3	-	-	$\mathbf{6.5 \times 10^{-2}}$	$\mathbf{3.7 \times 10^{-2}}$	$\mathbf{2.2 \times 10^{-50}}$
Model 4	-	-	-	5.4×10^{-1}	$\mathbf{2.1 \times 10^{-46}}$
Model 5	-	-	-	-	$\mathbf{2.2 \times 10^{-46}}$

In addition to that increase in score, the time needed by Model 2 to yield a prediction is half what is needed by Model 4. That is simply explained by the fact that due to the classification of the logs by length, the number of ICP comparison needed to yield a prediction is itself divided by two. Reducing the number of points in the scan also decreases significantly the prediction time, as can be seen from Models 3 and 5. Model 3 especially only need a little over a second to yield a prediction, i.e., a reduction by a factor 17 from Model 4.

Table 1 displays the Wilcoxon signed rank test p-values. Whenever a p-value is under 10%, it is possible to reject the $\mathscr{H}_0$ hypothesis and conclude that one of the models compared by the test yield more accurate predictions than the other. While we use here a confidence threshold at 10%, this could vary depending on the actual industrial process. First, it can be seen that Model 2 difference in score with Model 4 is statistically significant, as the p-value corresponding to the test between those two models is at 1%. On the contrary, Model 3 and Model 5 results, which both use voxel downsampling, can't be considered statistically different from Model 2 and Model 4 respectively. That is to say, any difference in score between those models isn't statistically significant when compared using the Wilcoxon test on our test set. Indeed, the p-values exposed in table 1 corresponding to the test between Model 2 and Model 3 is 12%, and the one corresponding to the test between Model 4 and Model 5 is 54%.

From all those results, we conclude our proposals lower the computational cost of using an ICP method to compare wood logs without significantly reducing the quality of the predictions. Some of those proposals even increase the average prediction/production score of the models. That is in particular the case when classifying the logs by length before running the ICP.

5. CONCLUSION

In this paper, we proposed two ways to reduce the computational cost of the multiple ICP runs needed to predict the basket of lumbers resulting from the sawing of a log. Our experiments highlight two main results. First, filtering the 3D scans of the logs and gathering logs by length considerably reduce both the individual runtime of an ICP and the number of needed comparisons. Second, the length clustering scheme and the replacement of a 1NN classifier by a kNN classifier improve the average score.

However, even with those improvements, using the ICP strategy to process unstructured data like the 3D scans remains less effective at predicting the basket of products than the much more classic Random Forest model.

An advantage of this approach is, nevertheless, its ability to be easily adapted to scans from any device, including terrestrial Lidars which are becoming increasingly common in the industrial world and produce high quality scans. In addition, both parametric and ICP methods could be mixed inside a classifier committee, as such methods are known to benefit from high diversity between their components.

REFERENCES

Besl, P.J. and McKay, N.D. (1992). A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2), 239–256.

Fix, E. and Hodges, J.L. (1951). *Discriminatory Analysis. Nonparametric Discrimination: Consistency Properties.*

FPInnovations (2014). *Optitek 10 User's Manual.*

Morin, M., Gaudreault, J., Brotherton, E., Paradis, F., Rolland, A., Wery, J., and Laviolette, F. (2020). Machine learning-based models of sawmills for better wood allocation planning. *International Journal of Production Economics*, 222, 107508.

Morin, M., Paradis, F., Rolland, A., Wery, J., Gaudreault, J., and Laviolette, F. (2015). Machine learning-based metamodels for sawing simulation.

Morneau-Pereira, M., Aa, M., Gaudreault, J., Nourelfath, M., and Ouhimmou, M. (2014). An optimization and simulation framework for integrated tactical planning of wood harvesting operations, wood allocation and lumber production.

Nasir, V. (2020). Wood sawing monitoring: sensory and artificial intelligence approaches. doi:http://dx.doi.org/10.14288/1.0389681.

Nguyen, V.T. (2018). Estimation de la qualité de bois ronds et d'arbres sur pied par lidar terrestre. (assessment of roundwood and tree quality from terrestrial lidar data).

Raczko, E. and Zagajewski, B. (2017). Comparison of support vector machine, random forest and neural network classifiers for tree species classification on airborne hyperspectral apex images. *European Journal of Remote Sensing*, 50(1), 144–154.

Selma, C., Haouzi, H.E., Thomas, P., Gaudreault, J., and Morin, M. (2017). An iterative closest point method for measuring the level of similarity of 3D log scans in wood industry. In *7th Workshop on Service Orientation in Holonic and Multi Agent Manufacturing, SOHOMA'17*. Nantes, France. Published in Service Orientation in Holonic and Multi-Agent Manufacturing, Borangiu T., Trentesaux D., Thomas A., Cardin O. (eds). Studies in Computational Intelligence, vol 762, pp. 433-444, Springer, Cham.

Thomas, E. (2017). An artificial neural network for real-time hardwood lumber grading. *Computers and Electronics in Agriculture*, 132, 71 – 75.

Wery, J., Gaudreault, J., Thomas, A., and Marier, P. (2018). Simulation-optimisation based framework for sales and operations planning taking into account new products opportunities in a co-production context. *Computers in Industry*, 94, 41 – 51.