# Real-time analysis of the grain on wooden planks

Michael Kellner and Tobias Hanning and Holger Farr

FORWISS, University of Passau, Innstrasse 51, D-94032 Passau, Germany

## ABSTRACT

A valuable visual indicator to grade the stiffness and strength of planks can be obtained by analyzing the structure of the grain on it. To integrate an analyzing image vision module in an industrial selection process a real-time system is needed to build. Two main objectives must be reached: First a stable edge detector should extract the grain edges. Second these grain edges have to be tracked to achieve a complete grain representation. This representation can be used to analyze the regularity of the grain. Since the visual nature of grain varies a lot even on a single plank we present an edge detector which is adaptive and a grain tracking algorithm capable of closing gaps between pixels. Both steps work in real-time (i.e. 5 frames per second resulting in 1 meter per second).

**Keywords:** grain analysis, strength grading, adaptive edge detection, grain tracking, real-time vision

## 1. INTRODUCTION

Strength and stiffness of wood are essential parameters for any further usage of lumber. Unfortunately these parameters can only be determined exactly by destructive methods. To avoid the destruction of lumber many visual features can be used for strength grading (cf.[1]).

New non-destructive methods to determine stiffness and strength are under research in the European project INTELLIWOOD (granted under no. IMS 29680). One of the main objectives of this project is to refine the relations between a vector of visual features and the stiffness and strength of the board.

For several reasons it is valuable to analyze the structure of the grain on the surface of wooden boards. In our case the reason was to conclude from the regularity of the grain to the stiffness and strength of the board. The grain is defined as the direction of the main anatomical elements of wood.

We designed and implemented a real-time capable vision system to measure the regularity of the grain. A suitable side-effect of our approach is that the result can also be used to classify the boards into appearance classes.

The crucial value for the grading of wood in this context is the so called modulus of elasticity (MOE). Once this value is determined planks can be graded into several quality classes. Nowadays the MOE can only be determined exactly by stretching and bowing the board with precise defined forces provided by very expensive measuring devices. Not only that such a measuring equipment is not affordable for many small and medium enterprises it is also a very tedious procedure, which should be spared in an industrial environment.

The connection between the regularity of the grain and the MOE of planks has been investigated in many publications (cf.,[2] [3] or[4]). Simplifying one can say, that if the grain is perfectly parallel to the main direction of the plank, the maximal strength of the wood is obtained (cf.[5]). So it seems natural to measure the difference to the ideal direction.

Although the regularity is a visual feature, in commercial products the regularity of the grain is measured by electrical capacitance (cf.[6] or the “slope-of-grain indicator” of Metriguard cf.[2]).

This article proposes a visual approach to measure the regularity of the grain on the top side of planks.

Further author information: (Send correspondence to M.K.)
M.K.: E-mail: kellner@forwiss.uni-passau.de
T.H.: E-mail: hanning@forwiss.uni-passau.de
H.F.: E-mail: farr@forwiss.uni-passau.de

Our setup was restricted to a common gray-value CCD-Camera, since empirical tests show that color information with its three times more amount of data to process is not in relation to the real-time restriction. The camera is located perpendicular to the plank at a distance of 1m.

The main direction of the grain can mathematically be written as the ordinate of the board, which should be also the ordinate of the image. For each row in the image we search for the grain lines, determine the main direction of each grain line and calculate the mean deviation to the ideal direction. To compensate outliers an average value is calculated over five up to ten pixel rows.

The main difficulties of this simple approach is to identify the grain pixels and to track the grain along the board over gaps between the pixels. The main constraint to any solution of these problems is the real-time capability. To solve the first problem we compared several edge detection techniques from simple Sobel filters to locally adaptive thresholding techniques. Since the grain has generally blurred edges, most of these standard techniques failed completely or delivered only insufficient results. In a first approach morphological methods were considered. Although these methods supply suitable results in most cases their runtime is too high. It turned out that a complete new approach, which combines a light model with an adaptive thresholding technique, matches best to the given problem. The new technique treats every pixel of a row only twice and detects up to 90 percent of the grain pixels.

To solve the second problem we developed a new grain tracking algorithm based on the Dijkstra algorithm to calculate cheapest paths in an egde-weighted directed graph. The weights on the edges will be calculated by an elliptic distance measuring the deviation of the next pixel to the main direction of the skein of pixels already tracked. Since this approach needs a lot of line fitting, we propose also an updating algorithm for the fitting of lines to sets of pixels iteratively enlarged by one pixel.

Altogether the whole algorithm is capable to handle up to 5 frames per second on a 1200 MHZ processor with a LINUX operating system. A whole board of 1m length can be examined in less than a second. The results are suitable to the given problem and promise to be a cheap alternative to the estimation of strength and stiffness of boards.

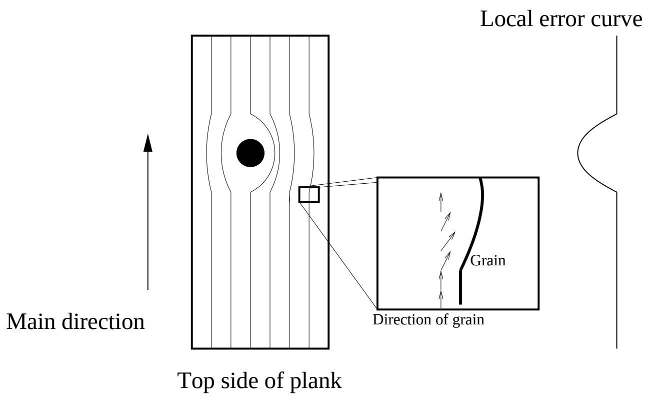

**Figure 1**: Sketch of the visual measuring approach

## 2. ADAPTIVE EDGE DETECTION

The first step in the analyzing process is to detect edges of wood grain. But as every natural phenomena grain edges vary a lot in their visual appearance. So simple thresholding techniques or edge detecting filters like sobel applied for the whole image must fail.

### 2.1. Local thresholding by n-level-set-fitting

An adaptive thresholding algorithm is the n-level-set-fitting presented in[7] or.[8] Dividing the image into a number of rectangles this algorithm can be applied for every rectangle providing an adaptive method to segment the

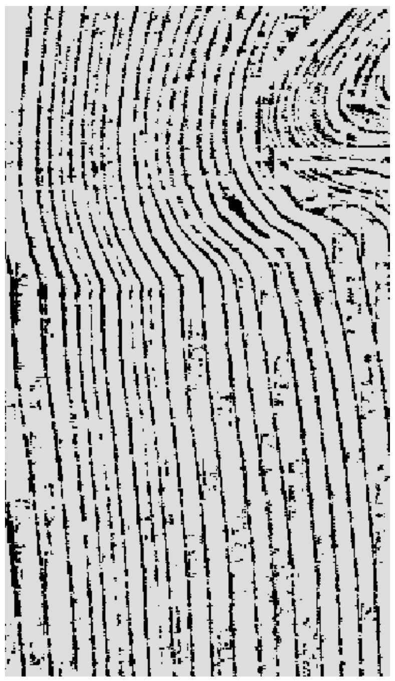

**Figure 2.** Grain edges detected by n-level-set-fitting: The image was divided in 10x10 rectangles. For each rectangle a 3-level-set-fitting was fulfilled and the darkest level displayed

darker grain from the brighter background. This proceeding turns out to be very stable, but it took about 2s to process the whole image.

The n-level-set-fitting provides a 2D thresholding approach, but in this context only a 1D approach is needed.

## 2.2. Line adaptive grain detection

A 1D approach to detect egdes has two advantages: First it is almost clear that it can be faster than a 2D approach, second it can also be used by a line scan camera setup, which is very common in visual wood inspection. We propose a line adaptive algorithm to detect local minima of the pixel line. Since the grain lines are mostly thin lines of a width of one up to five pixels the following algorithm **LINE ADAPTIVE MINIMA** leads to sufficient results:

Let $[p_0, \dots, p_{n-1}]$ be a pixel line. After approximating the data of the pixel line with a parabola $f$, set $P_f := \{p_i | i \in \{0, \dots, n-1\}, p_i < f(i)\}$ and let $P_0, \dots, P_{k-1}$ be the connectivity components in $P_f$. Then the algorithm **LINE ADAPTIVE MINIMA** calculates the global minima of each connectivity component in $P_f$ (cf. figure 3).

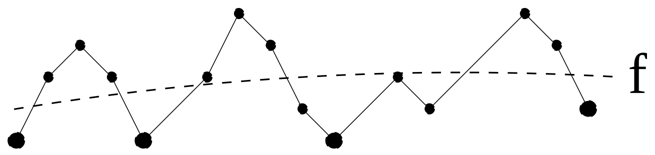

**Figure 3.** Sketch of an output of the algorithm **LINE ADAPTIVE MINIMA**: The circles mark the pixel values of a pixel line. The bigger circles mark the line adaptive local minima due to the algorithm

One can see the fitting of a parabola to a pixel line as modeling the lighting distribution of a spotlight over the board. Therefore the resulting pixels of the algorithm **LINE ADAPTIVE MINIMA** can be seen as adapted to such a lighting condition.

**Algorithm 1 LINE ADAPTIVE MINIMA** Calculates a set of local minima

**Input:** Line of pixels $[p_0, \ldots, p_{n-1}]$

**Output:** Set of local minima $(r_0, \ldots r_{j-1})$ in $1, \ldots, n$

```
1: Let $f(x) = ax^2 + bx + c$ such that $\sum_{i=0}^{n-1} (f(i) - p_i)^2$ is minimal.
2: $j := 0$
3: rem Check, whether we initially start below the parabola or not
4: if $p(0) < f(0)$ then
5:   $below$ := true;  $h := 0$;  $\min := p(0)$
6: else
7:   $below$ := false
8: end if
9: for $i := 1$ to $n - 1$ do
10:   if $below$ then
11:     if $p(i) < f(i)$ then
12:       rem still below
13:       if $min > p(i)$ then
14:         $min := p(i)$;  $h := i$
15:       end if
16:     else
17:       rem changed to above
18:       $below$ := false
19:       $r_j := h$;  $j := j + 1$
20:     end if
21:   else
22:     rem polygon lower than value at $i - 1$
23:     if $p(i) < f(i)$ then
24:       rem changed to below
25:       $min := p(i)$;  $h := i$
26:       $below$ := true
27:     end if
28:   end if
29: end for
30: return $(r_0, \ldots, r_{j-1})$
```

## 3. GRAIN TRACKING

The major difficulty to determine the main direction of the grain in the binary image, resulting from the algorithm LINE ADAPTIVE MINIMA, is to isolate the corresponding pixels from each other and from the distortion pixels. Since the grain contours in the binary image appear with a thickness of at most one pixel, due to the nature of the algorithm, every pixel belongs at most to one grain skein, except distortion pixels.

The proposed method tries to track every grain skein from an initial pixel to the end of the skein by using a method based on line fitting, dijkstra algorithm, elliptic distance and other basic techniques.

### 3.1. Graph Theory: Shortest Path

We define an undirected edge-weighted graph $G = (N, E)$, with nodes $N$ and edges $E$. A node $u \in N$ is chosen as the start node. The well-known Dijkstra algorithm[9] delivers a so called spanning tree of $G$. Each node is identified with a pixel and each edge represents the costs of connecting two pixels. For the described purpose, an elliptic distance is applied, also known as the mahalanobis distance.

Considering the resulting spanning tree, there is a path for every leaf to the root of $G$ with the minimum sum of weights of the edges along the path. Finally we have to determine the so-called *best leaf* (see 3.5), whose path represents the optimal grain skein on the plank.

### 3.2. The elliptic distance

Based on a directed straight line $L$ (determined by line fitting through the previously considered pixels) we define the elliptic distance as follows, whereas the fixed ratio $\alpha$ of the axes $a$ and $b$ of the ellipse is well-known.[*] With $b := 1$ the metrics $d_{ell}^{\alpha;n_1} : \mathbb{R}^2 \to \mathbb{R}$ is given by $d_{ell}^{\alpha,n_1}(p, p') := a$, i.e. the longer axis of the ellipse is taken as the distance value. To determine $a$ we assume that the point $p$ is the center of the ellipse and $p'$ fixes the ellipse. Since the normal form of an ellipse is used, $p'$ has to be rotated, so that the ellipse aligns with the given straight line $L$:

$$p"_1 =< t|n_2 > \text{ and } p"_2 =< t|n_1 > .$$

By solving $1 = \frac{p_1"^2}{a^2} + \frac{p_2"^2}{b^2}$ for $a$, the distance is determined depending on $\alpha$ and $n_1$.

Since we cannot assume, that the perpendicular foot $p \in L$ is known (see fig. 4), we have to determine $t$ which is given by $t = t_{\overline{p}} + (c, r)$, whereas $(c, r)$ is the vector between $p'$ and $\overline{p}$ (the current pixel)[†]. The displacement vector $t_{\overline{p}}$ is the vector from $\overline{p}$ to $p$, thus $t_{\overline{p}} = dist(\overline{p}, L) \cdot n_1$. Here $dist(\overline{p}, L)$ denotes the signed distance from the point $\overline{p}$ to the straight line $L$.

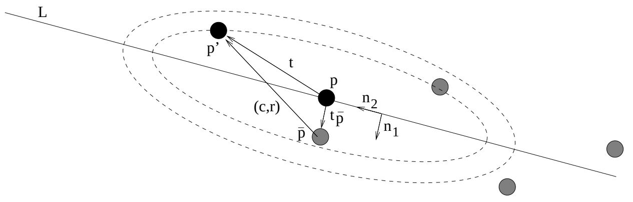

**Figure 4**: Calculation of the elliptic distance between $p$ and $p'$

Since the described metrics measures the distance to the points behind the current pixel as well, it could appear that the distance to such a pixel is smaller than a better one straight ahead. In figure 5 an example is depicted where $d_{ell}^{\alpha,n_1}(p, B) < d_{ell}^{\alpha,n_1}(p, A)$. Therefore the nearest pixel is $B$, which is obviously incorrect.

To solve this problem, all the edge weights to pixels not lying ahead the current pixel (the estimated direction is well-known in each step) are divided through a constant factor.

[*]Empirical good results can be achieved with $\alpha = 0.3$, respective $\alpha$ is calculated dynamically from the variances of the considered pixels, cf.3.3

[†]The notation $(c, r)$ is related to the relative column and row values in the distance map, described in 3.6

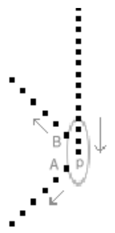

**Figure 5**: Problems with acute and obtuse angles

## 3.3. Line fitting

To determine the straight line $L_i$ a line fitting algorithm over the pixels $p_i, \ldots, p_{i+n-1}$ is used (practically the fitting is calculated on up to 5 pixels). Since we have to determine the $L_i$s in every step along the grain, the computation has to be very fast. An update strategy is useful, because $L_{i-1}$ fitted to $p_{i-1}, \ldots, p_{i+n-2}$ was determined in a previous step.

The simpliest way to describe a straight line is the slope-intercept form as follows $y = a \cdot x + t$. The minimizing problem can be expressed by

$$\sum_{i=0}^{n-1} (a \cdot x(i) + t - y(i))^2 \quad \text{minimal}$$

With differentiation by $a$ and $t$ we obtain the minima, by setting the terms to zero and solving the resulting equation by inserting into each other. So $a$ and $t$ depends only on the sums over $x_i$, $y_i$, $x_i^2$, $y_i^2$ and $x_i y_i$. At the step $i$ the update algorithm substracts the values of $p_{i-1}$ from the sums and adds those from $p_{i+n-1}$.

Naturally the result for the parameter $a$ can be very unstable, since all the $x_i$'s can be near to a vertical line. Therefore it makes sense to consider the variance $\sigma_X^2$ of all $x_i$ and $\sigma_Y^2$ of $y_i$, resp. If $\sigma_X^2 < \sigma_Y^2$ before calculating $a$ and $t$, we exchange the abscisse with the ordinate. So the computation error is quite small.

## 3.4. Implementation on binary images

The algorithm SKEIN TRACKER processes from a well-defined start pixel in a binary image through the neighbored pixels and thereby creates a set of pixel lists $S_i$. Every of these can be considered as a single skein. But after building these, a subset $S_i, \ldots, S_{i+n-1}$ is singled out and named the *best skein.* The choice strategy is described in 3.5. In figure 6 example skeins with all the possible pixel lists are depicted.

Let $HG$ be a hypergraph of the graph described above. A node is a list of pixels, a part of a skein already tracked. Edges are defined between nodes, if the startpixel of one pixellist is the beginning of an alternative way starting from a pixel in the other list. So $HG$ can be defined as a directed tree with a start node and leaves.

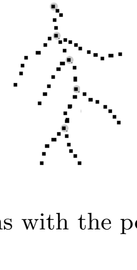

**Figure 6**: Skeins with the possible pixel lists

The both methods *list_add* and *list_begin_new_list* manage the pixel list structures. *list_add* adds the pixel to the current list and memorizes its distance to its predecessor. A list sorted ascending by the distance of all the currently last pixels of all pixel lists is built internally. The first pixel in the list is taken and returned. The method *list_begin_new_list* starts a new pixel list. So the behaviour of the shortest path algorithm mentioned above is accomplished.

**Algorithm 2 SKEIN TRACKER** tracks a skein on a binary image

**Input:** Binary image $f : \mathbb{R}^2 \rightarrow \mathbb{B}$, start pixel $r = (r_1, r_2)$, local search radius $rad$

**Output:** Lists of pixels $S_{i,n} = \{p_i, \ldots, p_{i+n-1}\} \subseteq S$ and $L \subseteq S$ a set of pixellists with no successors

```
p := r;
while is_valid(p) do
  set_marker_on(f,p);
  n := search_nearest_neighbor(p, rad);
  if is_valid(n) then
    d := dist(n,p);
    if is_set_marker_on(f,n) = FALSE then
      np := list_add(n, d);
    end if
  end if
  repeat
    n := search_next_neighbor(p, rad);
    if is_valid(n) then
      d := dist(n,p);
      if is_set_marker_on(f,n) = FALSE then
        list_begin_new_list(n, d);
      end if
    end if
  until (is_valid(n) = FALSE)
  p := np;
end while
```

## 3.5. The best skein

As already mentioned, the alternative ways through the defined graph are considered too. Since it can occur, that the tracking algorithm switches to some neighbored pixels, which itself represent an other skein, it is crucial to consider the alternatives (cf. figure 7). Many methods were tested, to determine the correct skein,

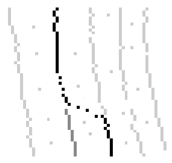

**Figure 7.** Tracking changes to a neighbor grain skein, since distortion pixels between the two contours lead to an error.

but a common strategy is not known yet. For every different application the algorithm is used, another choice technique is needed. In our experimental setup, where almost all grain skeins end up at the bottom of the plank, the following method is used:

Considering all the leaves in the mentioned hypergraph $HG$ above the one with the furthermost last pixel in the list to the first pixel in the start-node list is used.

It is clear, that other possibilities are given, e.g. the number of pixels along the lists from each leaf to the start node.

## 3.6. Time Speedup by Precalculating Distmaps

An ad-hoc implementation of the proposed algorithm leads to a very inefficient result, so that the real-time capability cannot be reached.

Due to the fact of working on discrete data, the speed can be increased, by using distance maps in which the current pixel is set to the middle of the map. With $R \subset \mathbb{N}$ und $M := \mathbb{N} \times \mathbb{N} \times \mathbb{R}$ a distance map $DM$ is defined as:

$$DM : R \rightarrow M \text{ whereas } DM(n) := (c, r, d)$$

The triples $(c, r, d)$ are sorted by $d$ ascending, whereas $c$ is the relative column, $r$ the relative row and $d$ is denoted as the corresponding distance value. So the expression $DM(0) = (c, r, d)$ describes the nearest pixel to the current, $DM(1)$ the second nearest and so on. By this way only a local region around each pixel is considered, so that each computation step takes only a constant amount of time.

The methods *search_nearest_neighbor* and *search_next_neighbor* in the algorithm 2 pass through the distance map, determine the pixel position in the image and check, if the corresponding flag is set.

To speed up the algorithm the distance maps are actually not calculated at the moment the next neighbor is searched. In an initialisation phase all possible distance maps were precalculated. The distance map depends on the size $m_1 \times m_2$, on the normal vector of the fitted line $n$, the ratio of the ellipse's axes and the translation vector $t_p$. Since we only can precompute a finite number of different distance maps, all the appearing values are quantised. So we can determine a unique index, with which we can look for the corresponding map in an array.

On every step nearly all the pixels in the image corresponding to the current map have to be considered. This could be restricted with some additional parameters: the maximum distance value $d_{max}$ and the maximum number of neighbors $neigh_{max}$. Since the maps are sorted by the distance, the algorithm can terminate when $d_{max}$ is reached. The branching nature of the algorithm is restricted by $neigh_{max}$, so that not all alternatives are treated.

## 4. EXPERIMENTAL RESULTS

### 4.1. Experimental setup

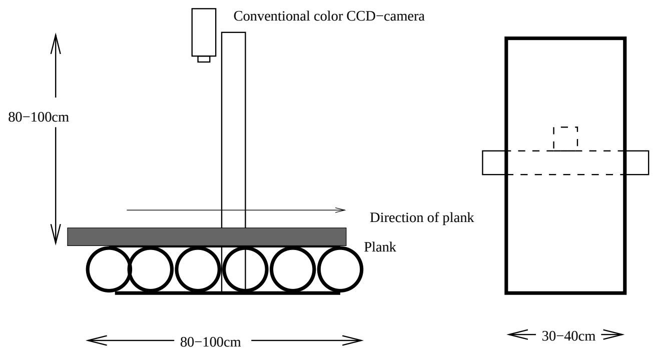

**Figure 8**: Sketch of the experimental setup

As depicted in figure 8 a common grayscale CCD-camera is used with a shutter time of 1/150. The camera is perpendicular to the board with a distance of about 100cm. The plank traverses with a speed of one meter per second. A cold light source without flickering is used not to disturb the high speed camera and obtain a diffuse light.

### 4.2. Results

On the pictures in figures 9 and 10 you can see the computation process. The binary image contains connected pixels representing the grain and some distortion pixels between them. Near the knot in image 9 the grain is blurred, which can cause problems other edge detection algorithms cannot cope with. Our approach extracts nearly every grain.

The skein tracker depicted in the next image only delivers bad results near the knot, since the curvature is very high. Practically the knots are detected otherwise, so that the influence can be eliminated further.

The graph on the right, represents the deviation from the main direction in every row. In this example the mean error value over 10 lines is ploted. By the way positive error axis shows to the right.

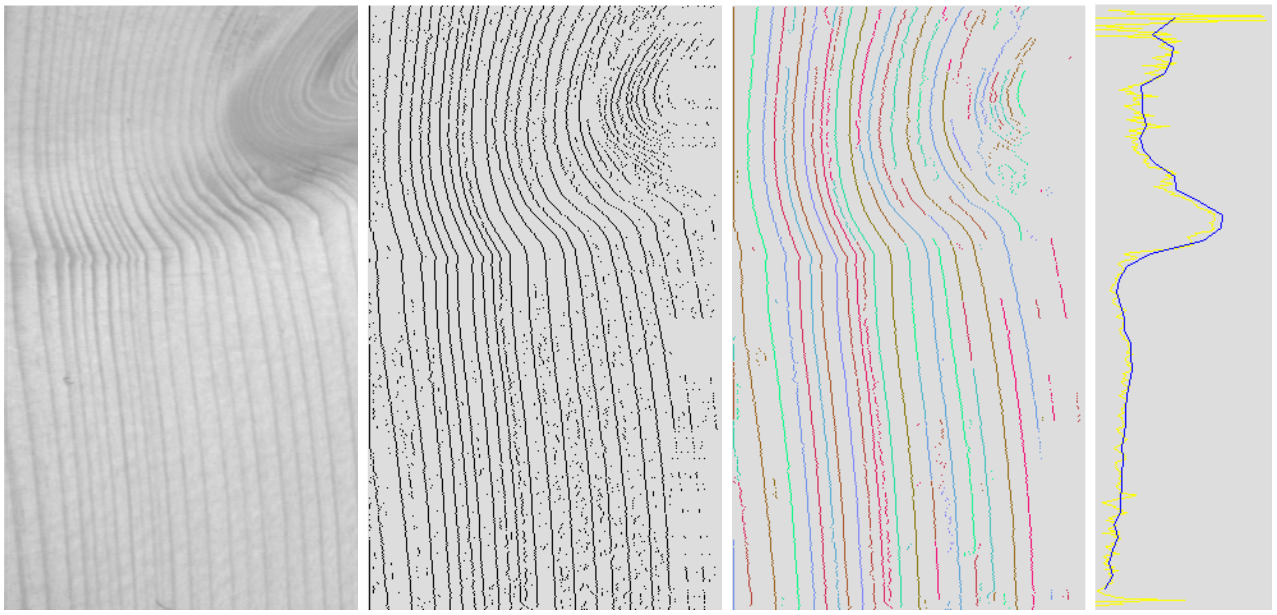

**Figure 9.** From left to right: the original capture of a board, the adaptive edge detection result (Algorithm LINE ADAPTIVE MINIMA), the skeins depicted in different grayscales and the deviation graph.

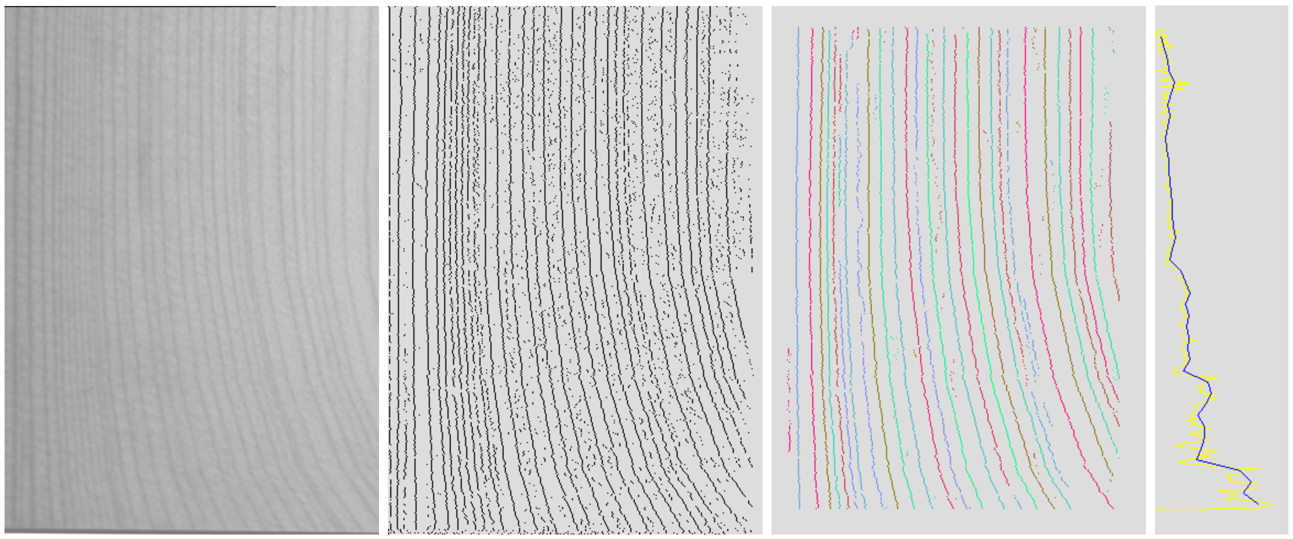

**Figure 10.** Another example, from left to right: original; result from the algorithm LINE ADAPTIVE MINIMA; result from SKEIN TRACKER; deviation graph.

## 5. CONCLUSION

A real-time capable implementation of an image processing method measuring the deviation of the grain from the main direction is presented to obtain an indicator to grade the stiffness and strength of planks. The results encourage to build a commercial product measuring the regularity of the grain by visual inspection. Since the regularity is also a visual feature this approach seems to be promising.

## ACKNOWLEDGMENTS

This work was supported by the European Union under Grant IMS 29680 (IntelliWood)

## REFERENCES

1. *Wood Handbook – Wood as an Engineering Material*, United States Department of Agriculture, Madison, WI., 1999. Gen. Tech. Rep. FPL-GTR-113.
2. M. Samson, "Measuring general slope of grain with the slope-of-grain indicator," *Forest Products Journal* **34**(7/8), pp. 75–78, 1984.
3. C. P. Heine, *Simulated Response of Degrading Hysteretic Joints With Slack Behavior.* PhD thesis, Virginia Tech, 2001.
4. D. W. Green and D. E. Kretschmann, "Lumber property relationships for engineering design standards," *Forest Products Journal* **23**(3), pp. 436–456, 1991.
5. ASTM, "Stand method for establishing structural grades and allowable properties for visually graded lumber," Tech. Rep. Standard D 245-81, American Society for testing and materials, Philadelphia, Pa., 1984.
6. K. A. McDonald and B. A. Bendtsen, "Measuring localized slope of grain by electrical capacitance," *Forest Products Journal* **36**(10), pp. 75–78, 1986.
7. T. Hanning, H. Farr, M. Kellner, and V. Lauren, "Segmentation of vector images by n-level-set-fitting," in *IEEE International Conference of Image Processing 2001*, (Thessaloniki), October 2001.
8. T. Hanning, "Nonparametric segmentation of grayvalue images by n-level-set-fitting," in *World Multiconference on Systemics, Cybernetics and Informatics 2000*, **V**, pp. 90–95, (Orlando), July 2000.
9. A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *Data Structures and Algorithms*, Addison-Wesley, 1983.