Project co-funded by the European Commission within the FP7 (2007–2013)

Grant agreement no.: 308630

I-PAN

INNOVATIVE POPLAR LOW DENSITY STRUCTURAL PANEL

Project type: Collaborative Project

Start date of project: 1st October 2012 Duration: 36 months

D7.2 – Report on strand forming technologies

WP n° and title	WP7 – Mat forming technologies
WP leader	UMIL
Responsible Author(s)	UMIL: Fabio Scotti
Contributor(s)	UMIL
Planned delivery date	M30 (March 2015)
Actual delivery date	M30 (March 2015)
Reporting period	RP2

Add an “x” in the box representing the dissemination level of the deliverable, as in the DoW

Dissemination Level		
PU	Public	x
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Document information

Abstract

An acquisition setup has been prepared to monitor the mat forming process, with special attention to the analysis of the orientation of the strands. This setup includes a camera located on top of the conveyor belt, together with a dedicated illumination system. Specific algorithms to analyze strand orientation based on computational intelligence were developed and tested. Also, a setup for the 3-D analysis of the strands was tested, to estimate the strand's shape and dimensions both from free-falling strands and from the mattress. The analysis of free falling strands is equivalent to the one developed for DL6.3. For the 3-D analysis of the strands laid on the conveyor belt, however, the results obtained indicate that this approach is not suitable for the analysis of the distribution and orientation of the strands in the mattress.

Keywords

Strand forming, orientation, 3D shape analysis, image segmentation, color clustering

Authors

Editor(s)	Fabio Scotti, Vincenzo Piuri, Andrea Valsecchi, Sergio Damas
Contributors	UMIL, ECSC
Peer Reviewers	IMAL

Document history

Version	Date	Reviewed paragraphs	Short description
0.1	*27/02/15*	*All*	*First draft*
0.2	*13/03/15*	*All*	*Second draft*
0.3	*25/03/15*	*All*	*Release for peer review*
1.0	*31/03/15*	*All*	*Final version for the EC*

* Abbreviations of editor/contributor name

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS AND DEFINITIONS

DM	Deliverable Manager
DoW	Description of Work
EC	European Commission
PM	Project Manager
PMQP	Project Management and Quality Plan
PMB	Project Management Board
PR	Peer reviewer
QM	Quality Manager
TMB	Technical Management Board
WP	Work Package

1 INTRODUCTION

The objective of this document is to introduce the hardware and software developed to monitor mat formation, with specific attention to strand orientation. The main objective is to ensure an appropriate orientation of wood strand in the mattress in order to achieve the desired panel properties. Three different visions systems have been studied:

1. A system that captures images of the wood strands during free-fall, and performs 3-D surface reconstruction from multiple views. This system permits to evaluate wood particle's shape and dimensions. The details of this system and the algorithms designed for it are discussed in *Deliverable 6.3 – Report on the elaborated engines and software for blending technology*.
2. A system that acquires images of the wood strands that have been laid on the mattress, and performs 3-D surface reconstruction from multiple views. This system uses the setup developed for the analysis of free-falling strands. However, the results obtained in the application tests show that this approach is not suitable for the analysis of the distribution and orientation of the strands in the mattress.
3. A system, named Belt2D, that consists of one camera placed perpendicularly to the conveyor belt where many the wood strands have been laid. The acquisition system is thus simple in structure, cost-effective and easy to deploy.

This document analyzes the results obtained using the third system. The aim is to process each of the 2D layers of the mat without requiring an expensive and time consuming 3D reconstruction. We show that the software is able to precisely measure the orientation of the wooden strands from a single image acquired by the system. In particular, this document focuses on the advanced image processing algorithms used to extract the orientation of the strands. We begin by reviewing the acquisition process and sketching the overall approach to be employed. Then, in Section 2, we introduce the background of the orientation extraction method. The complete technique is discussed in Section 3. In Section 4, we describe the experimental study performed to validate our software and discuss its outcome. The conclusions on the study are provided in Section 5.

1.1 LAYOUT OF THE VISION-BASED SYSTEM

This section describes the layout of the vision system of the prototype proposed by UMIL to monitor the mat-forming process. Two different acquisition setups have been projected. The main differences in their deployment is the placement of the illumination elements. Hereafter, the first acquisition setup will be called Belt2D and the second one, acquisition setup B. Belt2D is preferred to acquisition setup B. Hence, if the condition in the production plant permit it, Belt2D should be the one installed.

Figure *1* and **Figure *2*** show the layout of Belt2D from a side and top view, respectively.

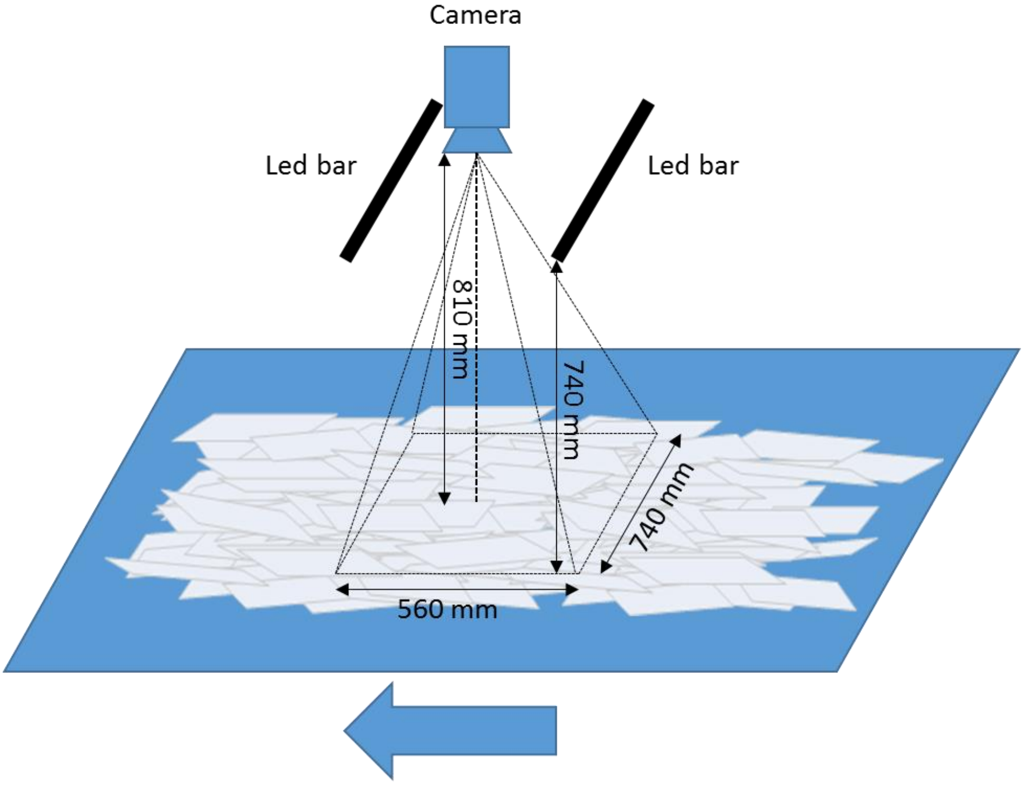

Figure 1 - Layout of Belt2D (side view). The arrow represents the direction in which the mattress moves.

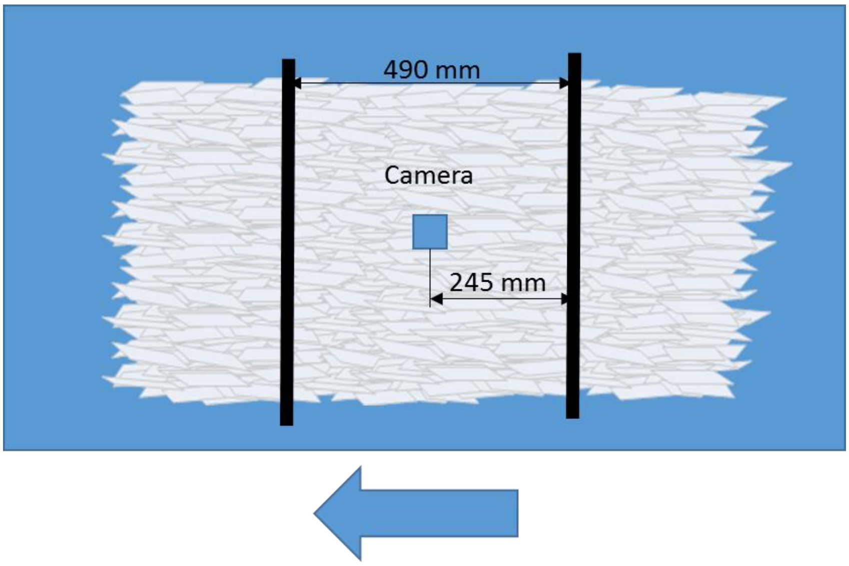

Figure 2 - Layout of the Belt2D (top view). The arrow represents the direction in which the mattress moves.

In case the deployment of Belt2D presents problems, acquisition setup B has been designed. The main difference with Belt2D is the orientation of the illumination elements, which are placed parallel to the movement of the conveyor belt. Figure 3 and Figure 4 show the structure of acquisition setup B from the side and top view, respectively.

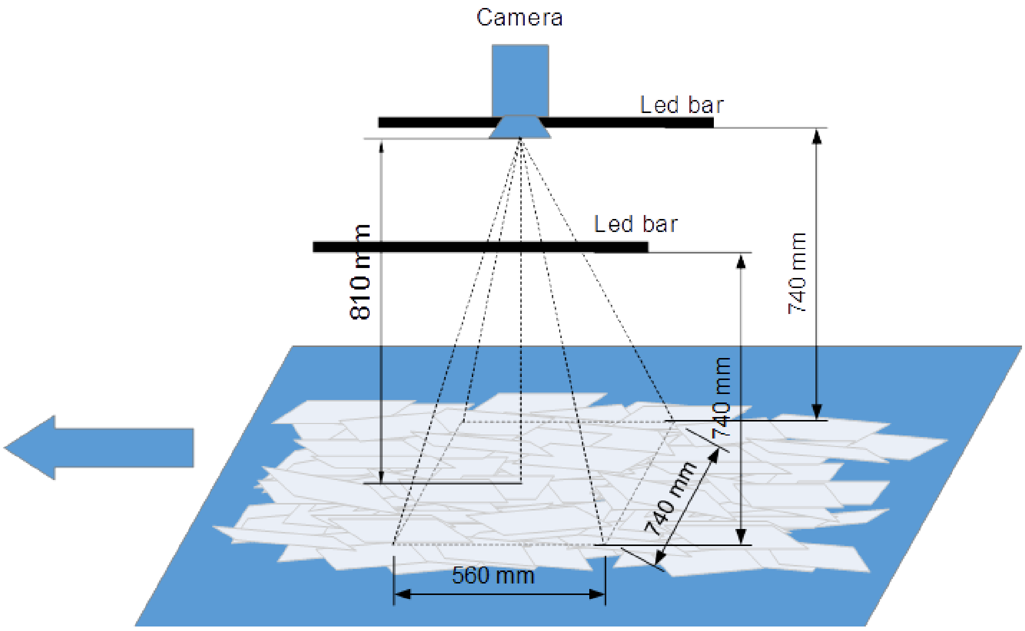

Figure 3 – Layout of the acquisition system B (side view). The arrow represents the direction in which the mattress moves.

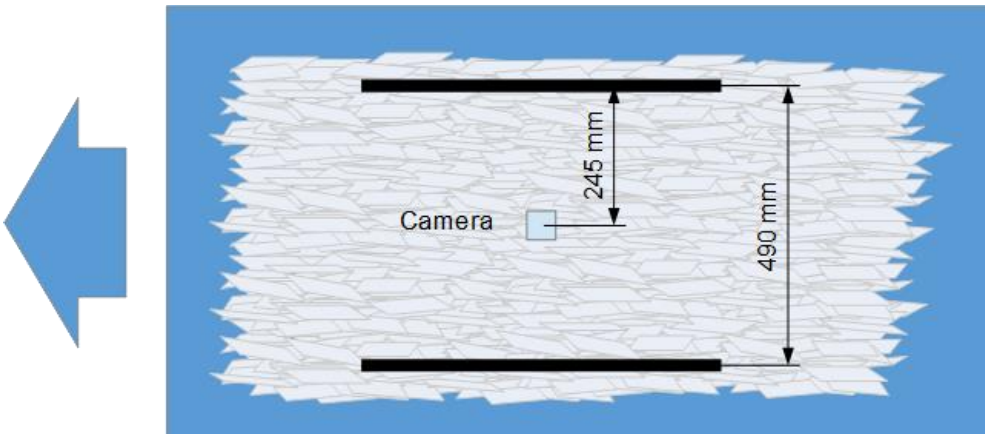

Figure 4 – Layout of the acquisition system B (top view). The arrow represents the direction in which the mattress moves.

1.2 Orientation extraction for Belt2D images

The acquisition setup is very simple. A Sony XCD-SX90CR CCD color camera is placed directly above the conveyor belt. Figure 5 shows an image acquired with this configuration.

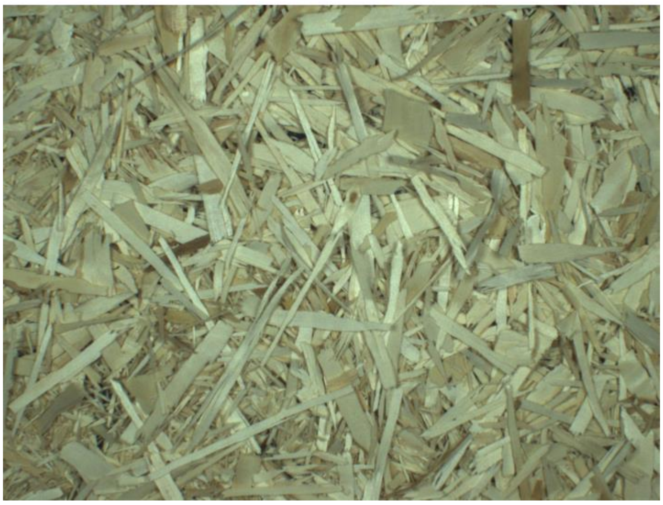

Figure 5 A Sony XCD-SX90CR CCD color camera (top). An example of image acquired with the setup (bottom).

The internal report on Strand Distribution Analysis (Fabio Scotti) contains a discussion on the general topic of detecting the orientation of wooden strands from image data. The document also introduces the topic of image segmentation. In brief, image segmentation is the process of partitioning an image into multiple regions, typically corresponding to certain objects or sharing some common characteristics, such as intensity, color or texture. Image segmentation is the main operation carried out in the orientation detection software. Indeed, the procedure followed by our method begins by performing a segmentation of the image into regions corresponding to individual strands. Then, the contours of these regions are used to compute the orientation of the strand. Alternative approaches that do not require segmentation, such as a simple analysis of the Fourier spectrum of the image (Nishimura and Ansell 2002), have been ruled out during the preliminary study because of their poor discriminating power.

Image segmentation is often based on detecting the contours of the objects to be segmented. In these cases, the first operation to be performed is the detection of *edges*, i.e. sharp variation in the intensity of the image. Figure 6 shows the result of applying a popular edge detection technique to Figure 5, the Canny edge detector (Canny 1986). Even with manual adjustments of the parameters of the method, the results are rather poor. One can spot the contours of largest strands, as they appear as large empty regions, but small strands are extremely difficult to tell apart. Most contours are incomplete or joined with those of other strands. As a consequence, the resulting segmentation is of very low quality.

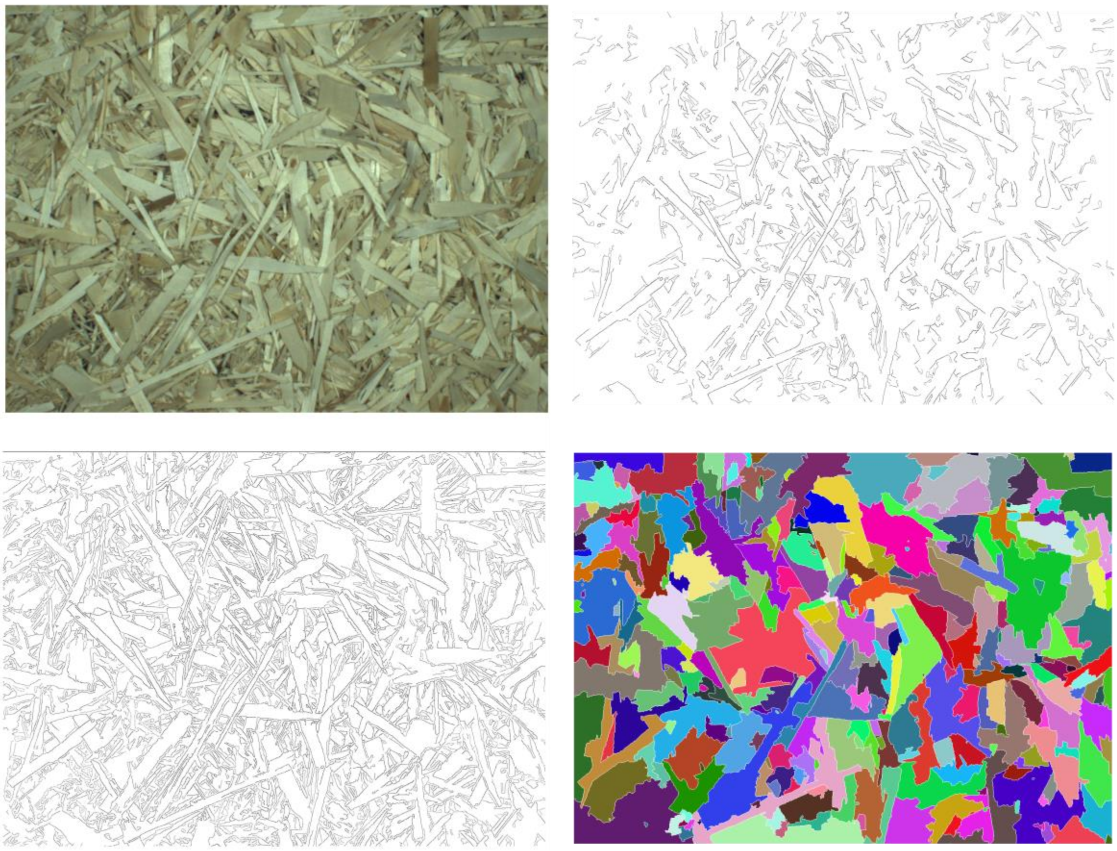

Figure 6 An example of the unfeasible results obtained using edge-based segmentation. Input image (top left), output of Canny's edge detection with default parameters (top right), output of Canny's edge detector after manual adjustment (bottom left), resulting segmentation (bottom right).

Overall, edge information is not reliable enough to be the basis of the segmentation. Instead, we turned our attention towards color as crucial property that will guide the segmentation process. While the range of colors in the image is rather small, strands tend to have a uniform tone. The color allows an observer to easily separate a strand from those around it. Our goal is to design an automatic segmentation method capable of performing this task as accurately as a human operator while saving time.

The key observation that originated our segmentation technique is the following. The pixels composing a strand have a similar position (they are not scattered across the image) and a similar color. The segmentation thus resemble the general problem of grouping some objects (in this case, the pixels of an image) according to their features (location and color) in such a way that objects in the same group are more similar to each other than to those in other groups. This task is known as *clustering* (Bishop 2006), and it is a major topic in data mining, machine learning, image analysis and information retrieval. Due to its relevance for the software, next section introduces clustering in general and its adaptation to color image segmentation.

2 BACKGROUND CONCEPTS

2.1 CLUSTERING

In the context of a clustering problem, an object is characterized by the values of a fixed set of F features. Therefore, objects can be thought as points in a F dimensional space. Figure 7 shows an example of data having just two features.

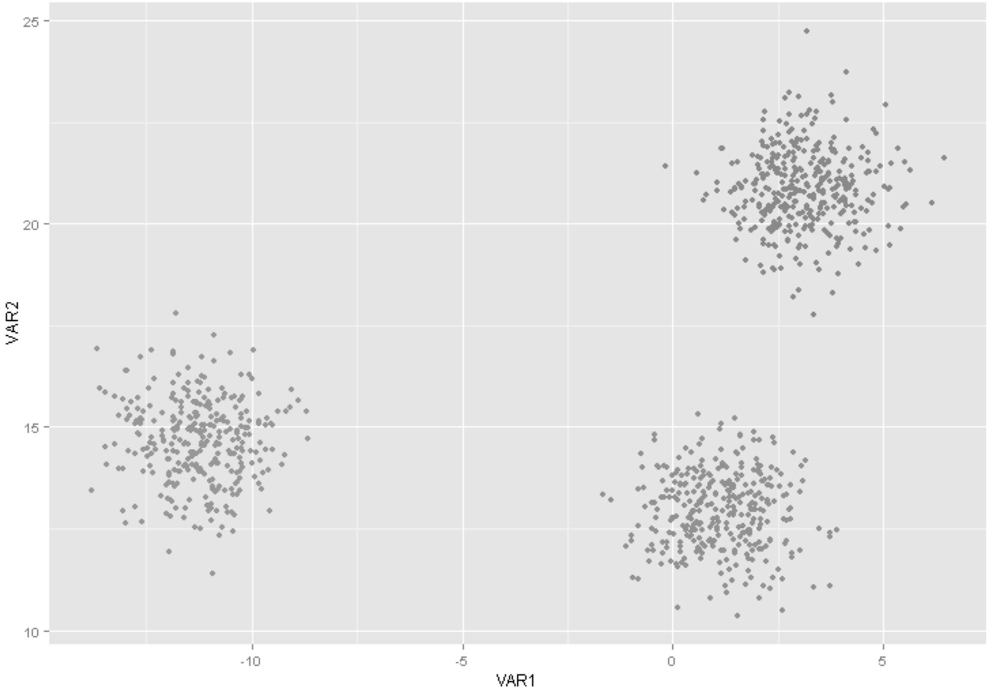

Figure 7 An example of data having two numerical features. Each data element (called object or observation) corresponds to a point on the plane.

A very intuitive way of clustering these data would be to create three circle-shaped groups, as shown in Figure 8. This is an example of centroid-based clustering (Lloyd 1982) : the groups are created so that data from the same cluster are close to each other, and data from different clusters are far from each other. There are though, many other criteria for clustering. Figure 9 show an example of clustering based on connectivity, i.e. the absence of large gaps between objects in the same group, regardless of location. Other clustering techniques use statistical analysis. In distribution-based clustering, one assumes the data is a mix of a specific set of statistical distributions, one for each cluster. This can be very powerful for the cases in which such assumption is justified. Finally, in density-based clustering (Kriegel et al. 2011), clusters are defined as areas of higher density than the remainder of the data set. Objects in these sparse areas, that are required to separate clusters, are usually considered to be noise and border points. No assumption on size or shape of the clusters is made. An example of this approach is shown in Figure 10.

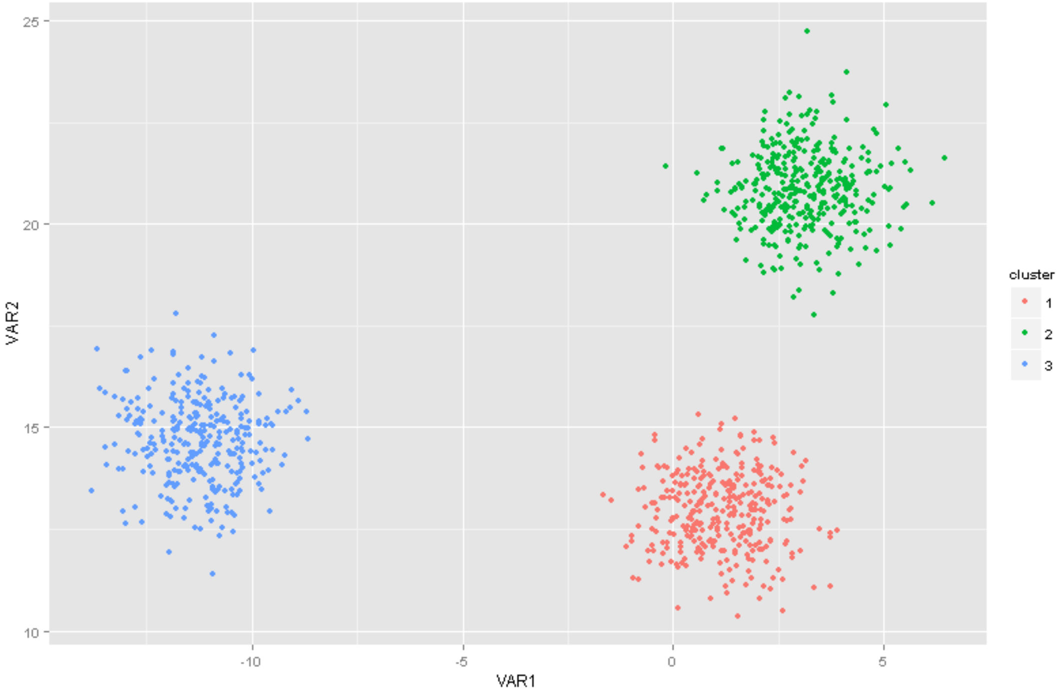

Figure 8 The outcome of centroid-based clustering applied to the example data from Figure 7.

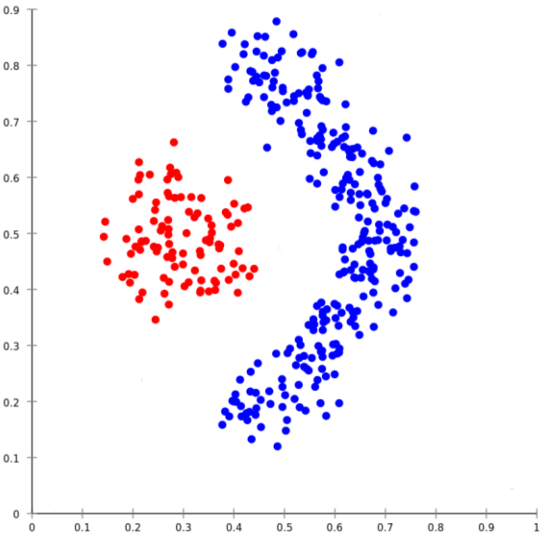

Figure 9 Clustering data based on connectivity. The gap between the two groups is the focus of the algorithm.

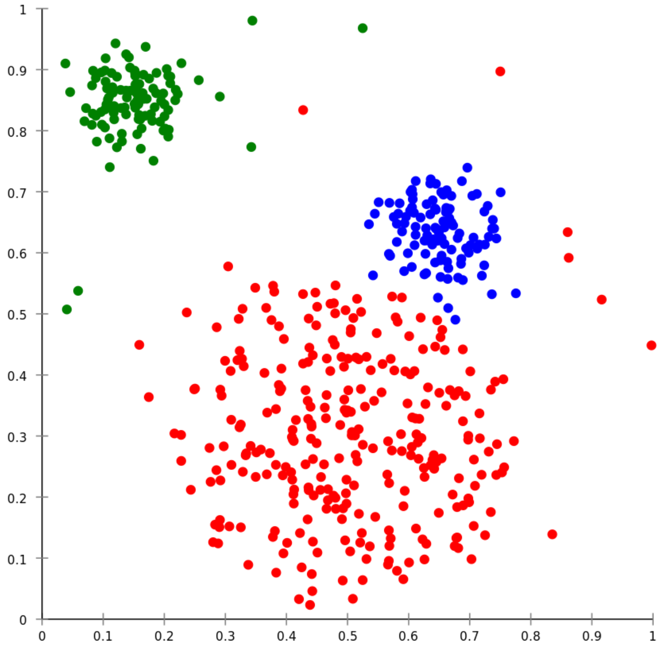

Figure 10 In density-based clustering, a change in density is what tell two groups apart, regardless of shape, size or location.

In general, there is no "correct" way to cluster a certain set of data (Estivill-Castro 2002), or even to determine how many groups there are. The purpose of the analysis and the contextual information about the data is what guides the choice of the clustering method. For illustrative purposes, we will introduce what is possibly the simplest and yet a common clustering approach: the k-means algorithm (Lloyd 1982).

2.1.1 K-MEANS CLUSTERING

The k-means clustering method is the reference centroid-based clustering algorithm. Given a set of objects $(x_1, x_2, \dots, x_n)$, k-means clustering aims to partition the n elements into k (≤ n) sets S = $\{S_1, S_2, \dots, S_k\}$ so as to minimize the within-cluster mean distance. In other words, the algorithm splits the data in k distinct groups such that the distance between each element and the center of its group is as small as possible. The center (or mean position) of the i-th cluster S_i is

$$\mu_i = \frac{1}{|S_i|} \sum x \in S_i x$$

Within S_i, the mean distance between the center and a point is

$$D_i = \frac{1}{|S_i|} \sum x \in S_i | |x - \mu_i| |$$

The objective of k-means is to minimize the sum of these distances across the whole clustering, i.e. to make as small as possible the sum

$$\sum_{i=1}^{k} D_i$$

Starting from k random mean positions, the algorithm proceeds iteratively by alternating between two steps. In the assignment step, each object is assigned to the cluster having the nearest mean position. In the update step, the mean position of each cluster is set to the current centroid. The algorithm has converged when the assignments no longer change.

Note that the number of clusters k is an input parameter of the algorithm. This is the main limitation of k-means: an inappropriate choice of k may yield poor results. Another key limitation of k-means is that the clusters are expected to be elliptical (in feature space) and of similar size. In a number of situation, this is highly unsuitable. Next section will provide an illustrative example.

2.2 Clustering for image segmentation

In image segmentation, clustering is mostly used for color-based segmentation (Forsyth and Ponce 2002). In a typical application, the objects to be clustered are the pixels of the image, and their features are the color components (for instance, expressed as a combination of red, green and blue values). An overview of the process is shown in Figure 11, while Figure 12 shows an example of this kind of segmentation. Note that the segmentation is purely based on color information, thus a segment can be disconnected. i.e. made up of two or more isolated areas.

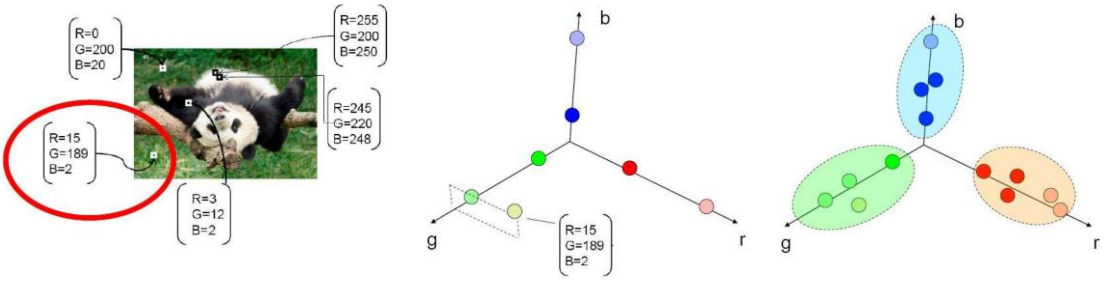

Figure 11 The process of segmenting an image by clustering the colors. Each pixel is represented as a point in the RGB color space. The resulting set of data points is then clustered. In the final segmentation, two pixels belong to the same segment if the corresponding colors belong to the same cluster.

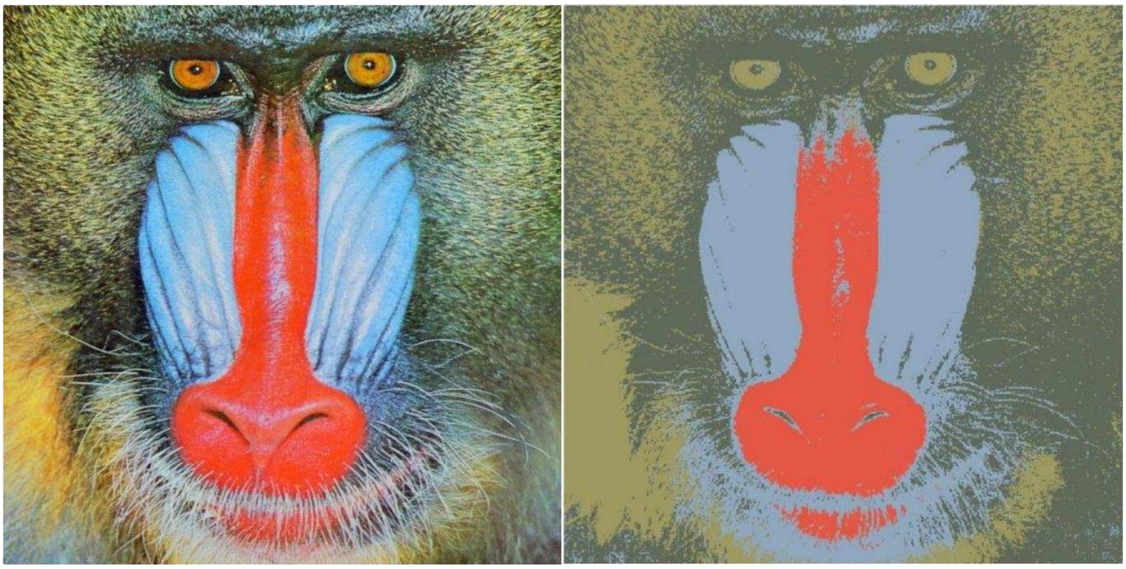

Figure 12 A segmentation based on color information only. The resulting segmentation has four segments made of many disconnected parts.

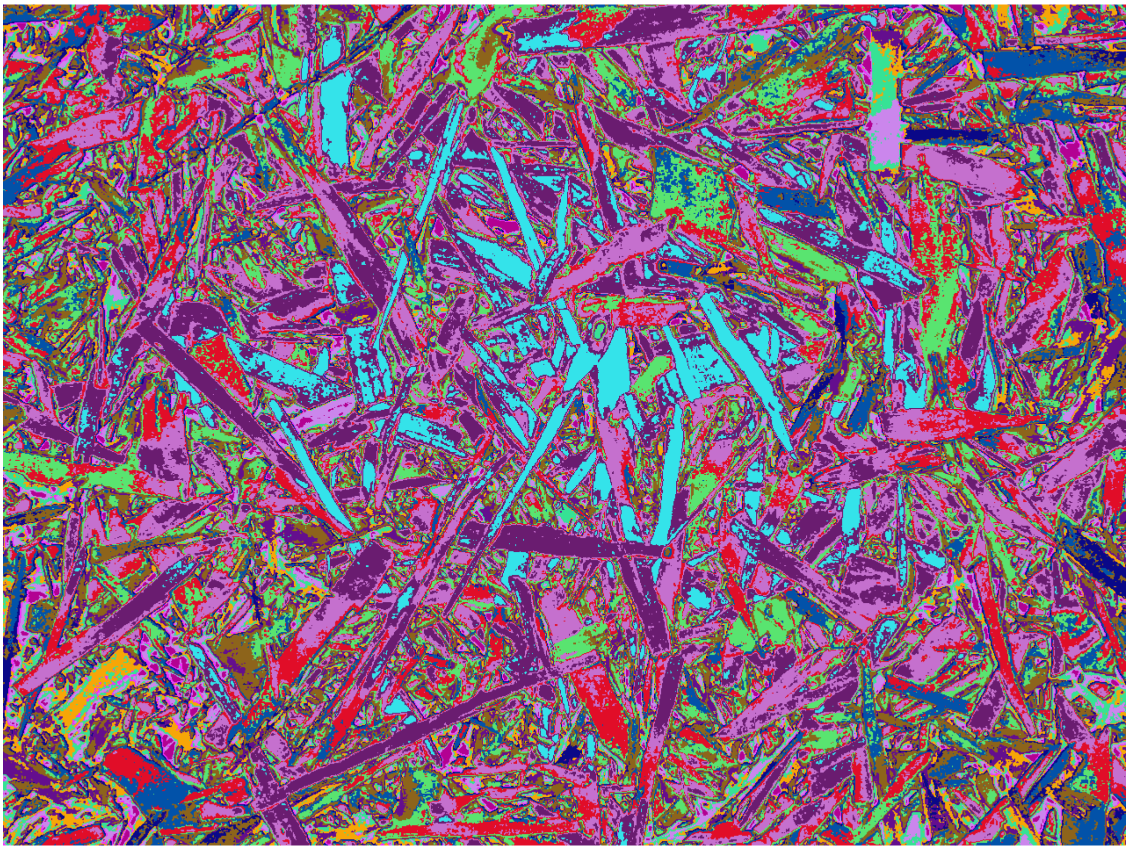

Figure 13 Color information only is not effective enough in segmenting strand images.

Alternatively, one can use the position of a pixel, together with its color, as features for the clustering. This results in a segmentation in which areas that are far from each other are put into different segments even if they have the same color, such as in the example in Figure 14.

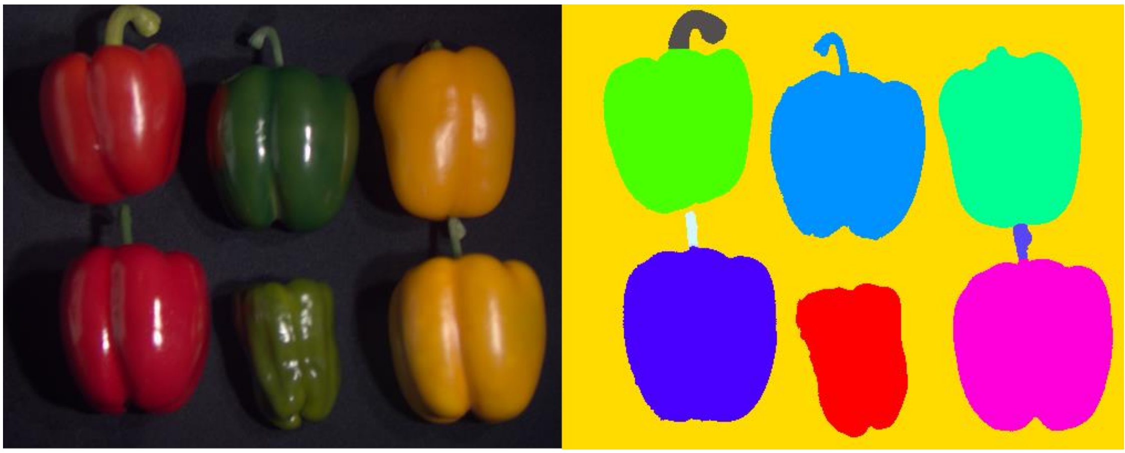

Figure 5 Segmentation based on clustering color and position information. Area with approximately the same color (e.g. two red peppers) are told apart by their difference in position.

This was the overall ideas we sketched in Section 1. However, the choice of the clustering algorithm is also key. For instance, k-means is not appropriate for the task, as it tends to create clusters having similar size in feature space, while the size of the strands can vary greatly even across the same image. In what follows, we introduce a density-based clustering algorithm that does not suffer from this limitation.

2.2.1 THE MEAN-SHIFT SEGMENTATION ALGORITHM

The mean-shift segmentation method (Comaniciu and Meer 2002, Arbelaez et al. (2011)) is the adaptation of a general-purpose clustering algorithm to color image segmentation. For simplicity, we begin by introducing the mean-shift clustering method and then continue with the detailed description of the actual segmentation method.

2.2.1.1 Mean-shift clustering

This method belongs to the family of density-based clustering algorithms. The overall idea is to determine the densest areas of the data (i.e. the modes) and to group together all the objects that lie in the "attraction basin" of the same mode. This basing is determined through an iterative process called the mean-shift procedure (Fukunaga and Hostetler 1975).

The algorithm applies the mean-shift procedure to every object in the dataset and track which mode is found at the end of the procedure. In the final clustering, each cluster corresponds to a single mode and vice versa. A cluster C is formed exactly by the objects from which the mean-shift procedure ended in the mode m_C corresponding to C. Figure 15 shows a simple example of how the clusters are created.

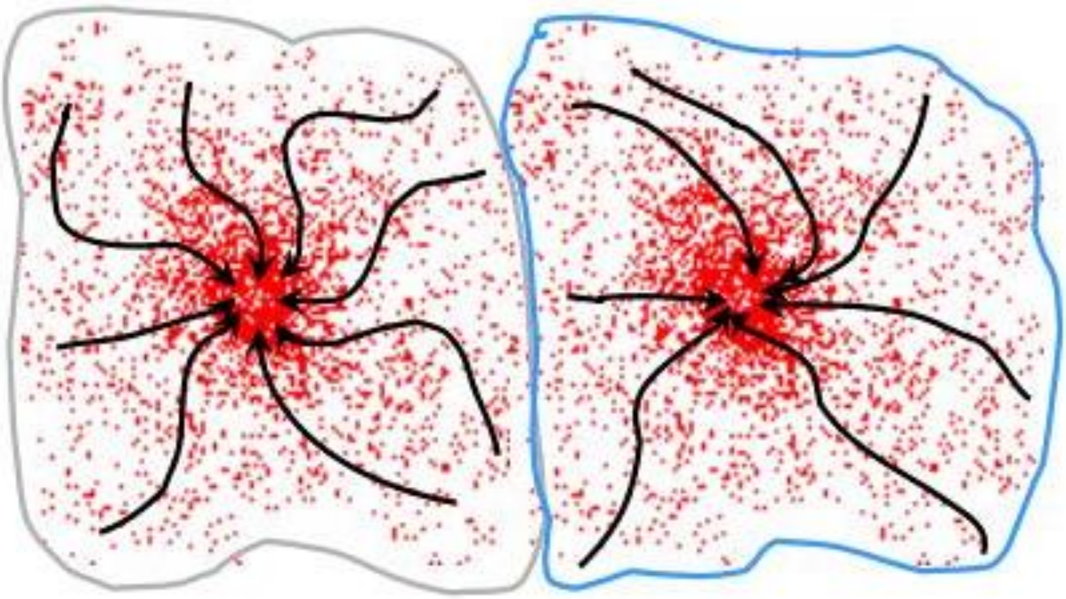

Figure 6 A dataset having two modes. The black arrows sketch some of trajectories of the mean-shift procedure, starting from a data element at the tail and ending in one of the two modes. The resulting attraction basins have been outlines in gray and blue.

Starting from an initial data point p as current position, the mean-shift procedure operates as follows.

1. Consider the spherical region of radius h around the current position
2. Compute the center of mass m of the region
3. If m is equal to the current position
   - Flag m as a mode
   - Assign p to the cluster C_m corresponding to m
   - End the procedure
4. Consider m as new current position and go to step 1

A graphical depiction of the first two steps is provided in Figure 16.

Note that the algorithm depends on just one parameter. The meaning of the parameter h is analogous to the bandwidth parameter in kernel density estimation, a statistical method to compute the underlying density (or equivalently, probability) distribution of a dataset. The larger the bandwidth, the smoother is the resulting density function.

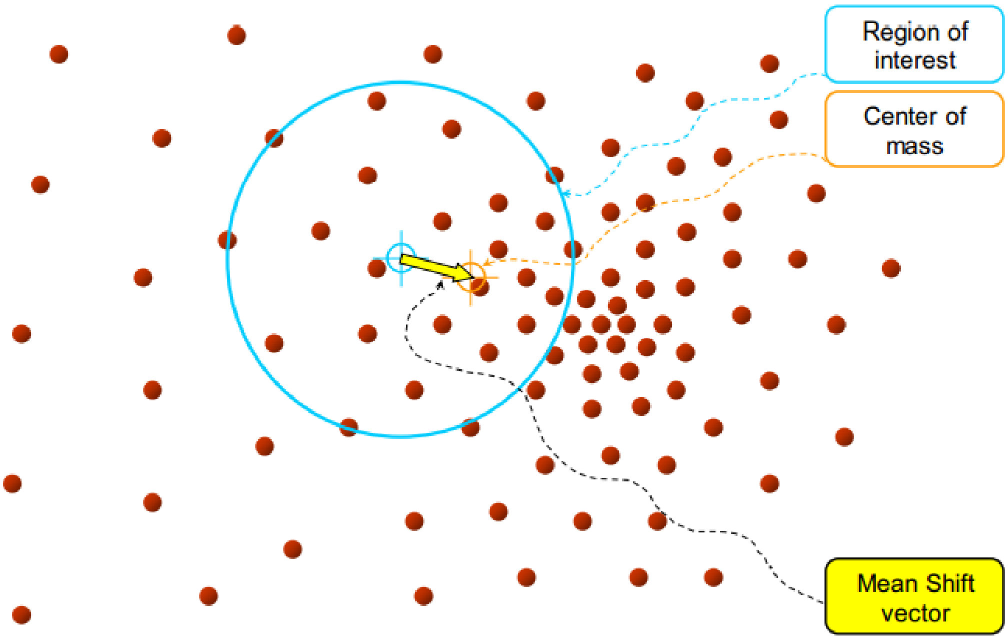

Figure 7 One iteration of the mean-shift procedure.

Despite the relatively simple calculation involved, mean-shift clustering can exhibit a very complex behavior; see Figure 17 for an example.

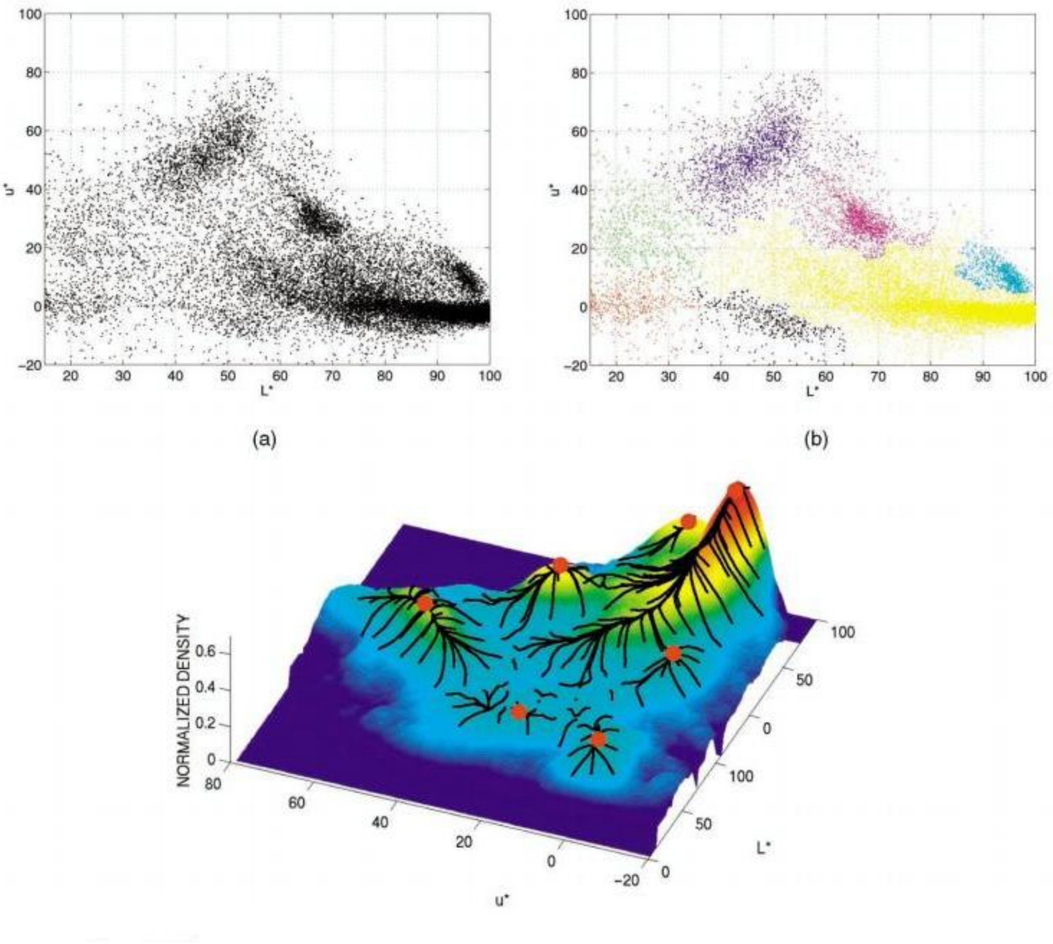

Figure 8 Mean-shift clustering applied to a complex dataset (top). The density estimation of the input data (bottom). The modes are marked with a circle. Note the variability in shape and size of the clusters.

2.2.2 MEAN-SHIFT CLUSTERING FOR IMAGE SEGMENTATION

As expected, the mean-shift segmentation method operates by applying the mean-shift clustering algorithm over a dataset containing position and color information of every pixel. There is, though, a key difference: instead of a single radius h there are two radiuses, one for color and one for position. This provides the ability to weight differently the magnitude of the change in color and position across a cluster, allowing the user to decide the relative importance of the two features over the segmentation.

3 ORIENTATION EXTRACTION

In this section, we introduced the proposed software method for measuring the orientation of wooden strands in Belt2D images. Figure 18 shows the overall structure of the method. At the outermost level there are two operations:

1. The contour of each wooden strand is detected by segmenting the corresponding area of the image.
2. The orientation of the wooden strands is computed from their contours.

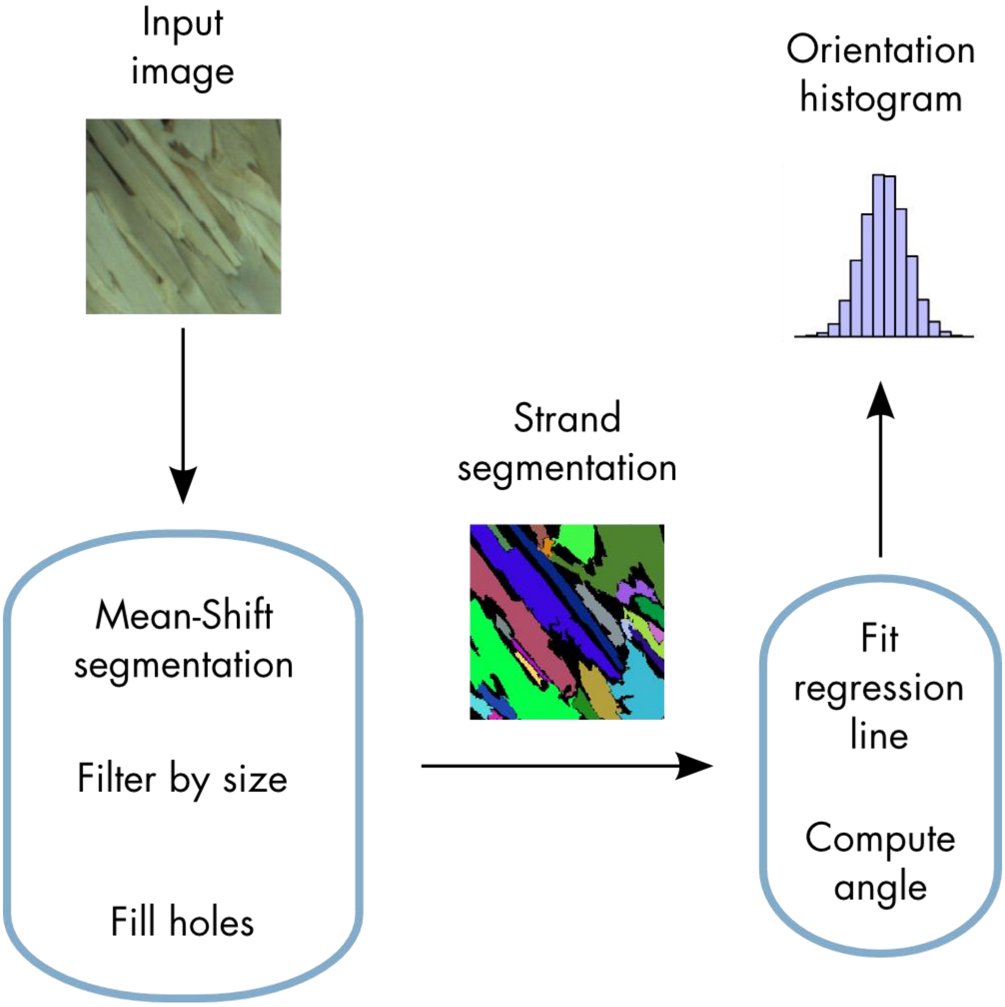

Figure 9 The overall structure of the orientation detection method.

3.1.1.1 Segmentation of the wooden strands

The segmentation process begins by applying mean-shift clustering to the pixels of the image based on position and color. The color of a pixel is specified by the three red, green and blue components, while the position is simply the row and column number in which the pixel appears. Figure 19, top right, show the raw output of this step applied to an example strand image. The colors are assigned randomly so that the individual strands are easy to spot. Note that the algorithm usually detects a very large number of extremely small regions, corresponding to strands that are almost completely covered by others. These regions are then removed based on their size, and the corresponding pixels are set to background and ignored. A small region which is completely contained into another is considered a hole of the latter, rather than one strand being on top of the other. Thus, these holes are filled with the surrounding label in the last phase of the algorithm (Figure 19, bottom row).

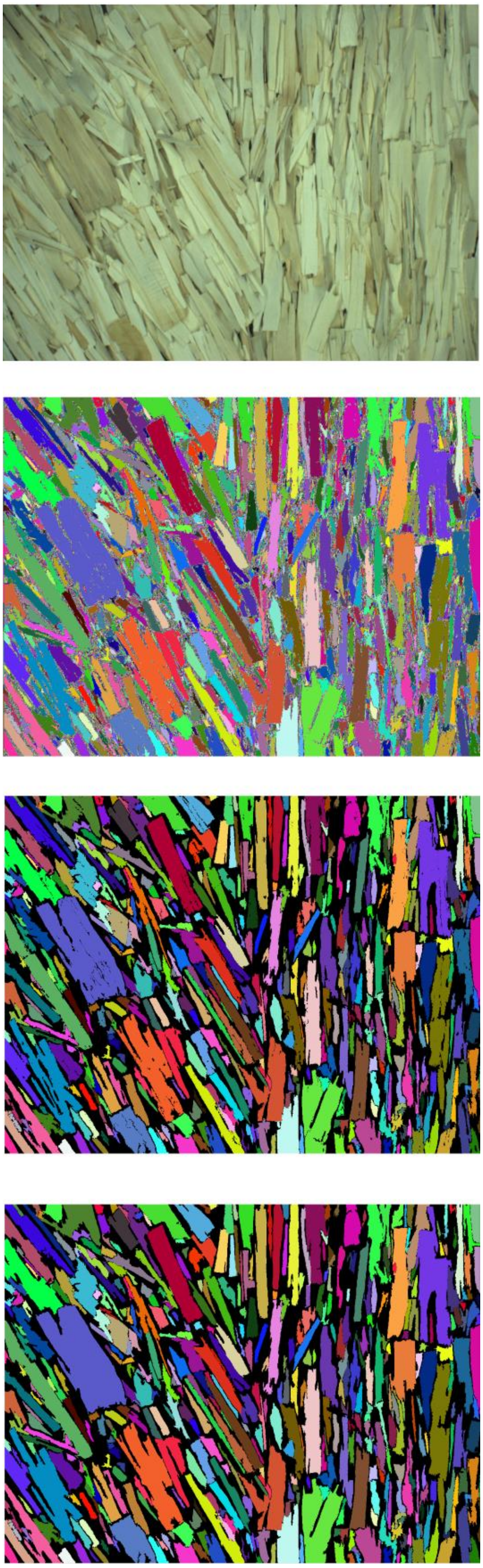

Figure 19 The image processing performed by the orientation detection software. The input images (top) undergoes mean-shift segmentation (second image). Then, the smallest regions are filtered (third image) and internal holes in other regions are filled (bottom).

3.1.1.2 Computing the orientation from the contour

Due to the production process at the plant, a strand is of approximately rectangular shape and its orientation is that of its major axis. A straightforward approach would be to compute a bounding rectangle around the contour of the strand and use the major axis. However, this is not very robust to irregular shapes or mistakes in the detection of the contour. Instead, we use least square fitting (Hastie, Tibshirani, and Friedman 2001), a procedure analogous to computing a regression line over a set of points.

The orientation of a strand is computed as follows. We consider the set of points P falling inside the strand contour C. Then, we compute the line the least square regression line r of P. The line r is such that distance with respect to the points in P is a small as possible. Recall that a line can be written in the form $\mathrm{y} = \alpha\mathrm{x} + \beta$ and let $\mathrm{P} = \{(\mathrm{x}_1, \mathrm{y}_1), \dots, (\mathrm{x}_\mathrm{n}, \mathrm{y}_\mathrm{n})\}$. The distance between a point (x, y) and the corresponding point on the line is $|\mathrm{y} - (\alpha\mathrm{x} + \beta)|$. The least square regression line is such that

$$\sum \mathrm{p} \in \mathrm{P}(\,\mathrm{y}_\mathrm{p} - (\alpha\mathrm{x}_\mathrm{p} + \beta))^2$$

is as small as possible. Figure 20 provide an example. Finally, the orientation angle is computed from the coefficients of the line.

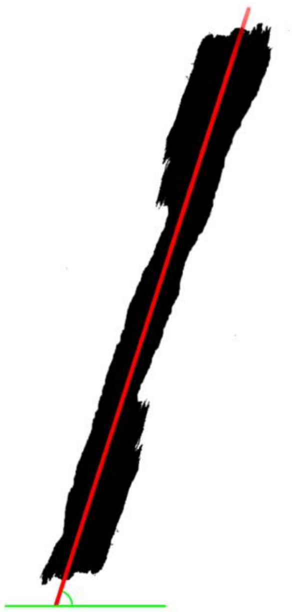

Figure 20 The computation of the orientation angle from a strand contour (in black) using least square regression line (in red).

4 EXPERIMENTAL STUDIES

After its design and development, the software has been tested in two experimental studies. The first is a preliminary study, involving test images acquired with a low quality camera. The experiment allows to check the effect of image quality over the accuracy of the software and possibly to confirm the specifications on the image acquisition system. In the second study, instead, the images have been acquired using proper equipment and lighting conditions. The wooden strands have been arranged according to a predetermined orientation distribution. This allows to perform a quantitative evaluation of the results of our orientation detection technique, thus validating the performance of the software. Both datasets were created at UNIMI simulating the camera setup and lighting of the factory. Each experiment is reported in a separate section.

4.1 PRELIMINARY EXPERIMENT

4.1.1 TEST SETUP

The images used for this study have been acquired using a cheap webcam. The dataset is composed of 26 images having size 640x480 pixels. A closer look at the images reveals a noticeable amount of color artifacts. An example is shown in Figure 17.

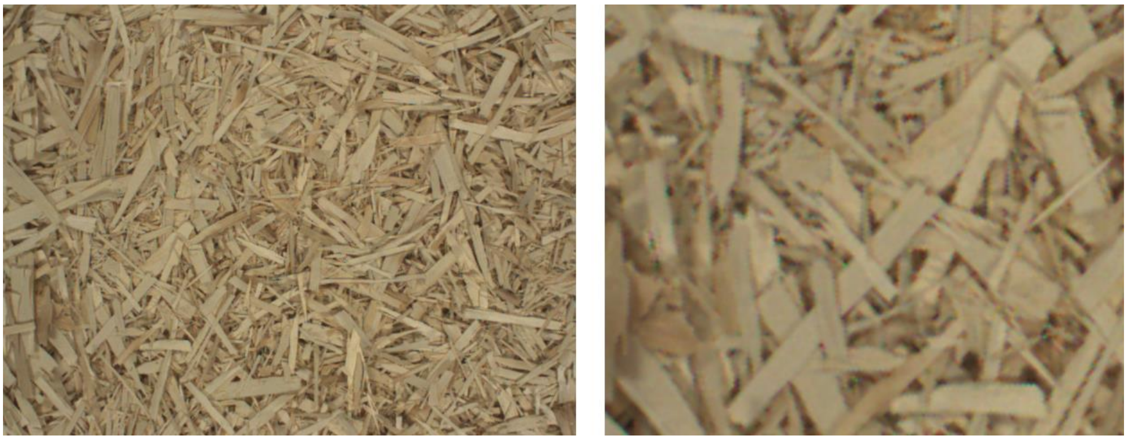

Figure 10 An image used in the preliminary experiment (left). A close up showing chromatic noise (right).

The parameters of the algorithm are three: the color and spatial radius for the mean-shift segmentation and the minimum strand area for filtering. Given the nature of this experiment, we tested the software extensively, using many parameters values, to make sure that potential inaccurate results are not due to an incorrect configuration of the algorithm.

4.1.2 RESULTS

Figures 18, 19 and 20 show a three of the segmentation obtained by the algorithm using the best configuration of parameters.

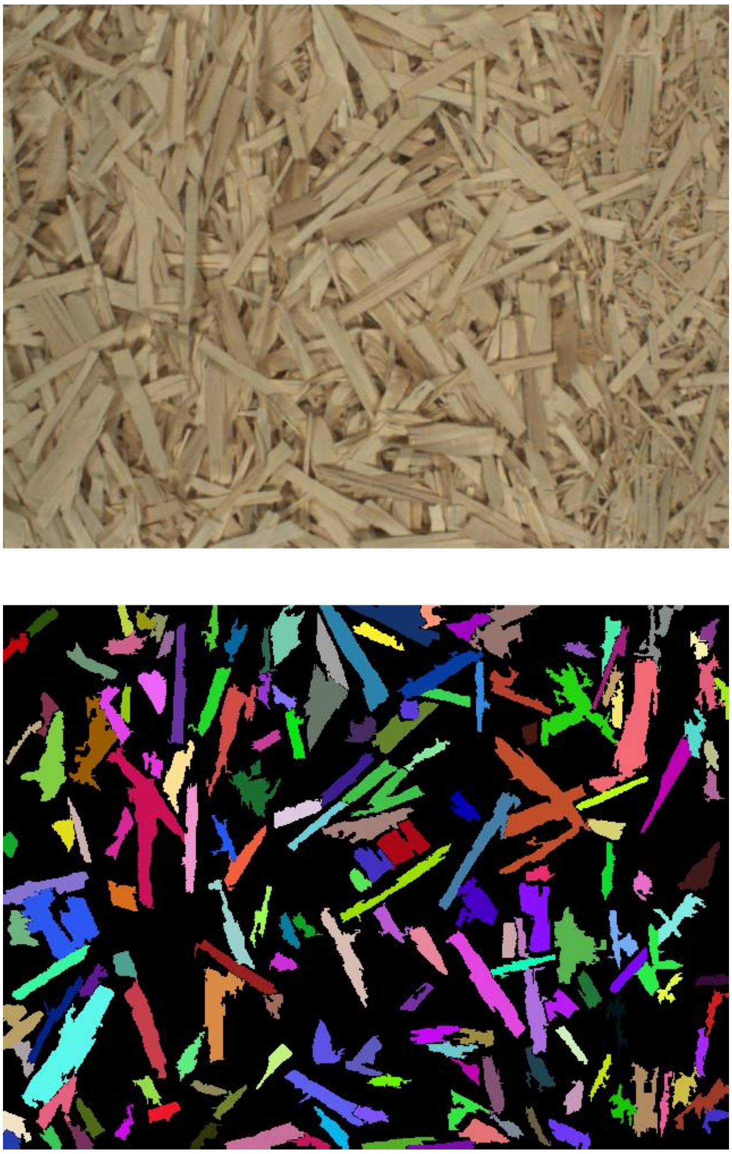

Figure 11 Preliminary experiment: a conveyor belt image acquired through a low quality camera (top) and the corresponding strand segmentation as performed by our software (bottom).

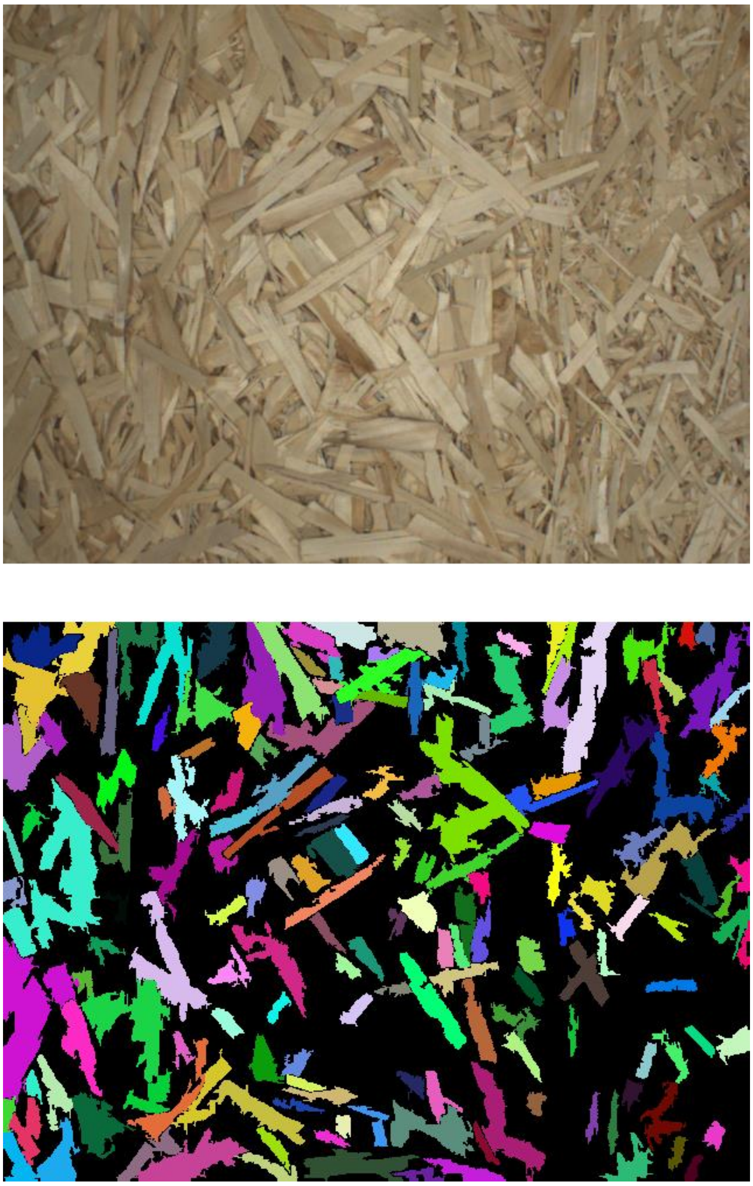

Figure 12 Preliminary experiment: a conveyor belt image acquired through a low quality camera (top) and the corresponding strand segmentation as performed by our software (bottom).

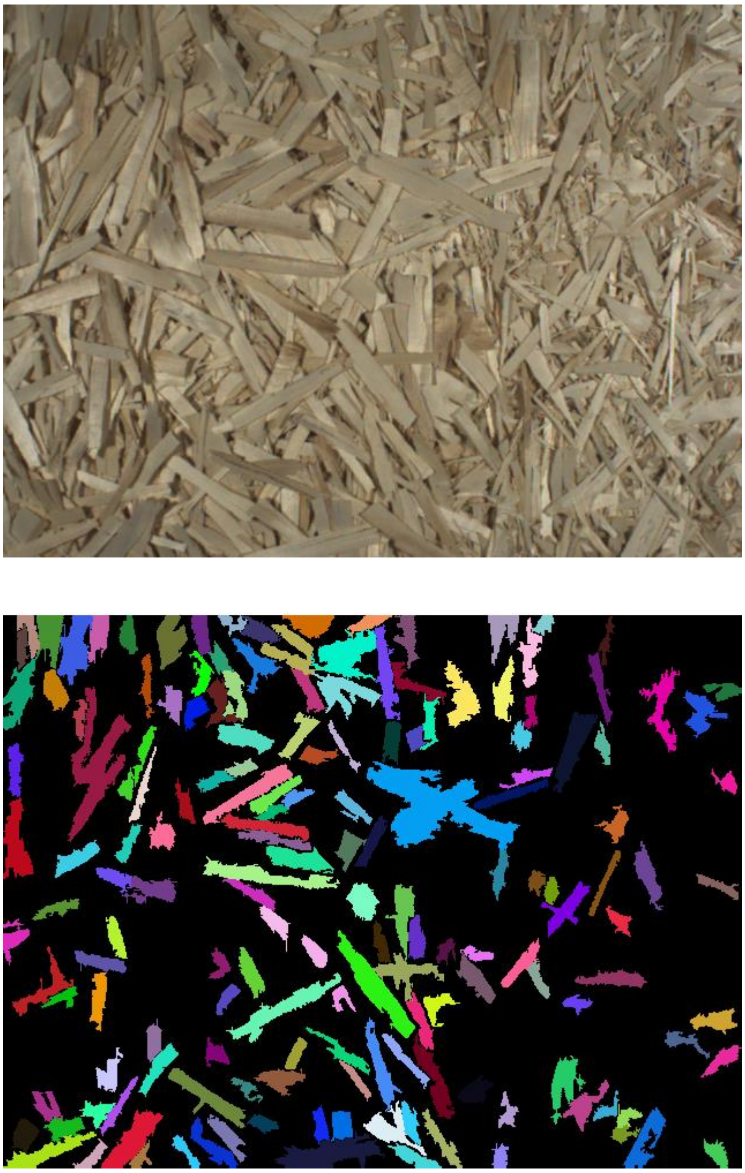

Figure 13 Preliminary experiment: a conveyor belt image acquired through a low quality camera (top) and the corresponding strand segmentation as performed by our software (bottom).

The accuracy of the segmentation is quite poor. While most large strands have been segmented correctly, most small strands have not been detected. Also, in a number of cases two or more strands have been joined together. As a consequence, the computed orientation angle will be incorrect.

The results of the preliminary experiment show that the combination of image acquisition system and software is not able to measure accurately the orientation of the strands. Through the second experimental study, we will show that this outcome is due to the low image quality delivered by the camera rather than because of a problem in the software component.

4.2 Validation experiment

4.2.1 Test setup

The images used in this experiment are meant to reproduce as accurately as possible the images from the panel production environment. Thus, they have been acquired according to the specification of the image acquisition system, using the Sony XCD-SX90CR CCD color camera and the camera setup called Belt2D. The dataset is composed of seven images. In each image, the strands exhibit a specific distribution of orientation angles that allows to evaluate the results of the software not only by looking at the segmentation, but also at the estimated orientation distribution. Figure 21 shows one of the images in the dataset. The parameter values used in the experiment have been determined with a few tests, simply by trial and error.

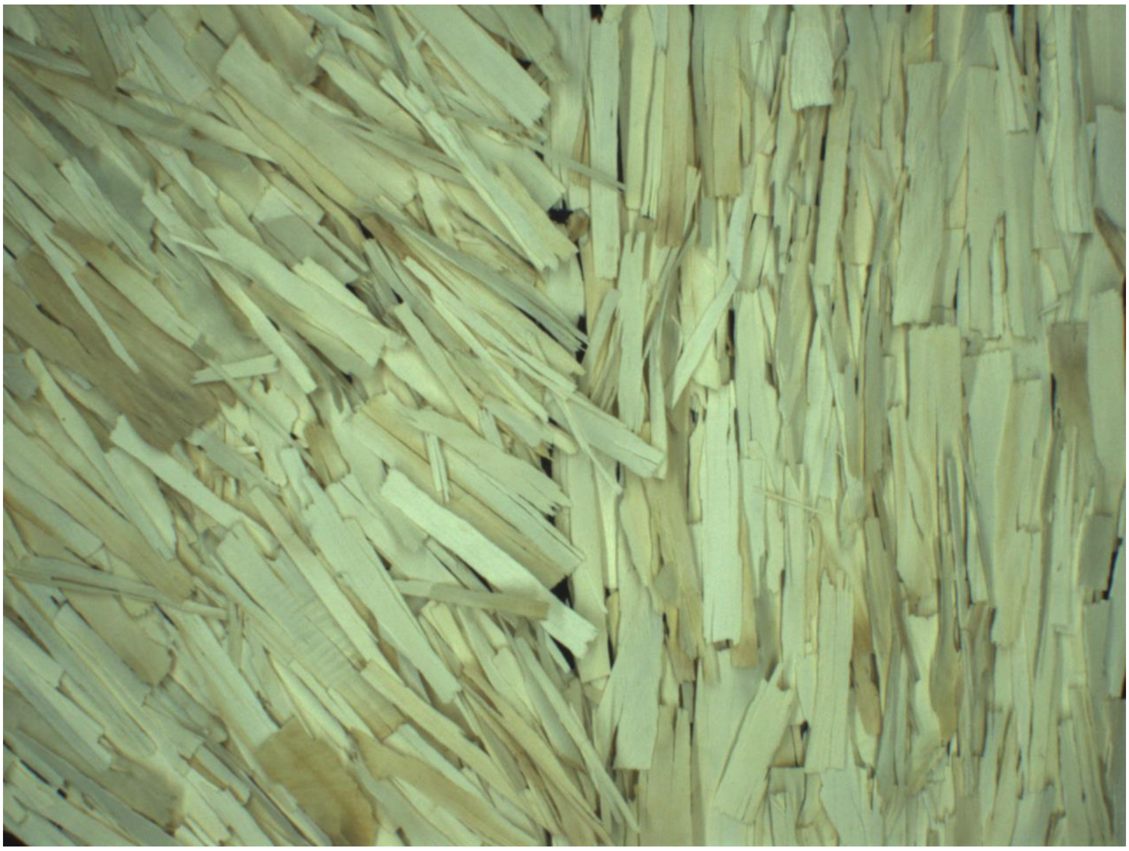

Figure 14 An image from validation dataset, having a predetermined orientation distribution. The strands in the right part of the image are oriented according to a 90° angle, while in the left part the orientation is 135°.

4.2.2 Results

Figures 22 - 28 show the seven test images, together with the corresponding strand segmentations and the histograms of the orientation. In blue there is the histogram computed by the software, while in red is the expected distribution.

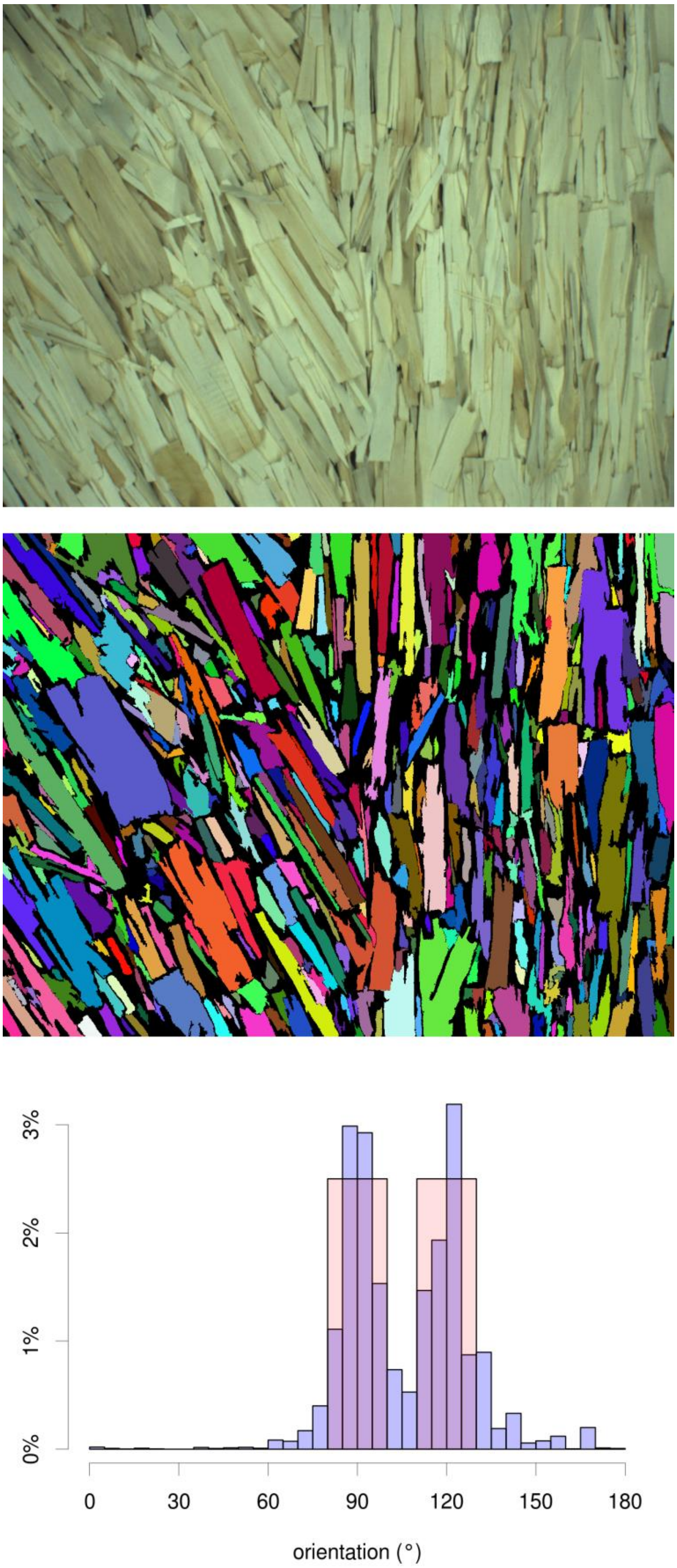

Figure 15 Validation experiment. Input image (top), strand segmentation (center), histogram of the strand orientation (bottom). In blue, the histogram as computed by the software; in red, the expected result.

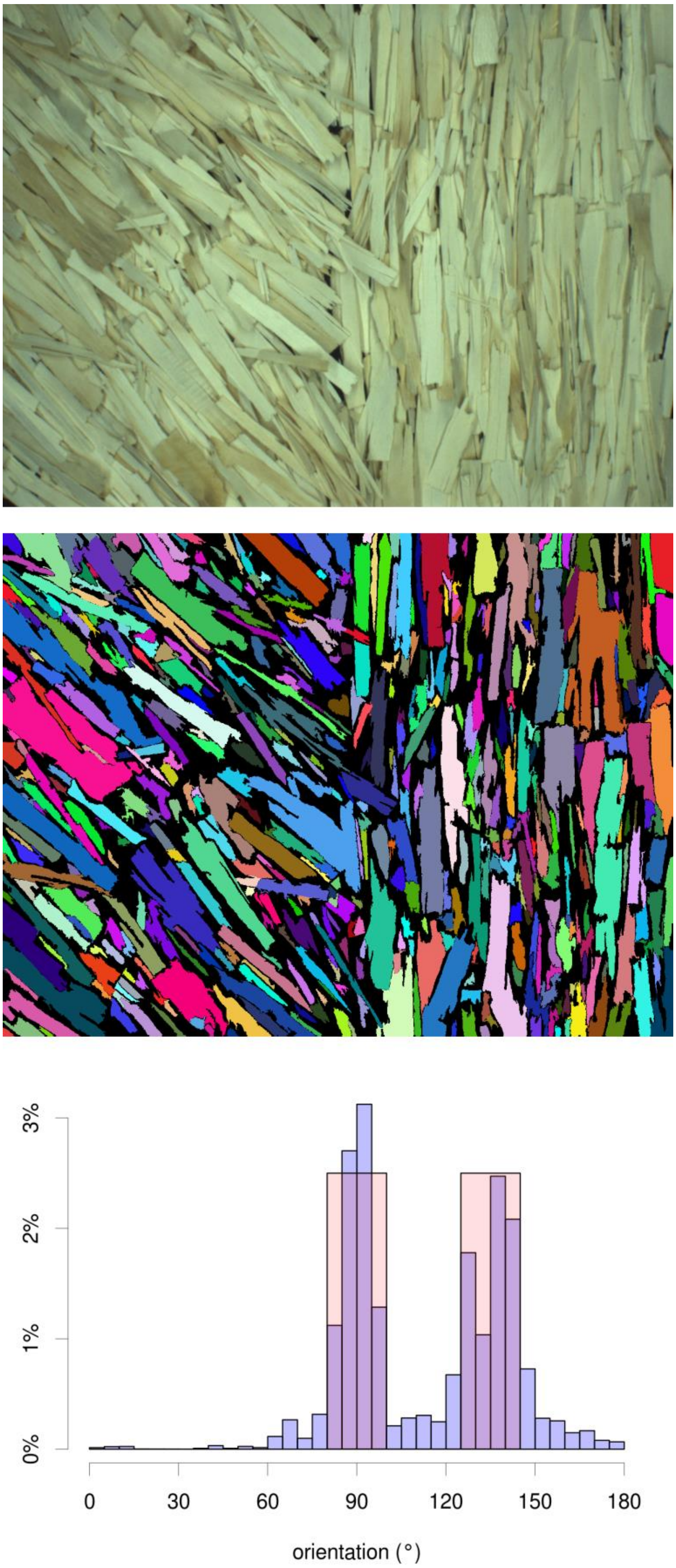

Figure 16 Validation experiment. Input image (top), strand segmentation (center), histogram of the strand orientation (bottom). In blue, the histogram as computed by the software; in red, the expected result.

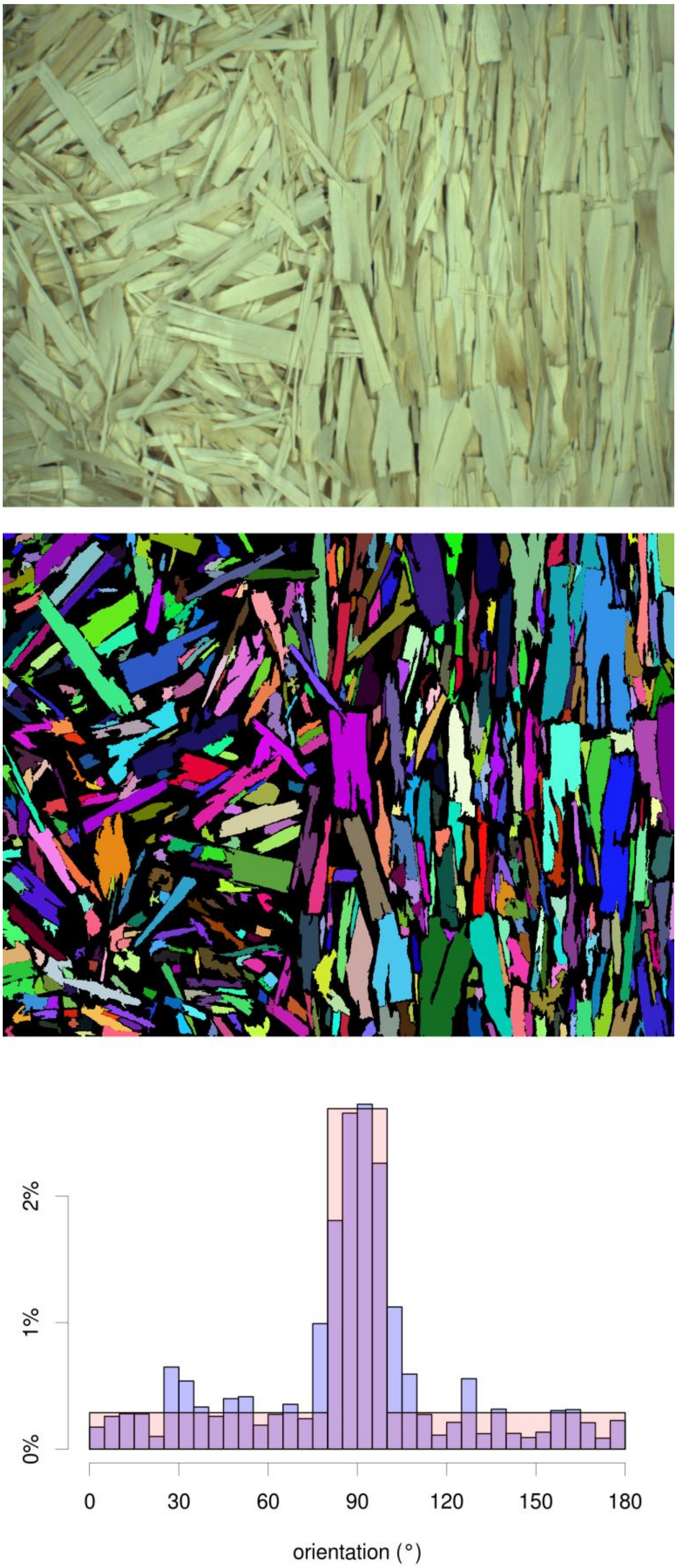

Figure 17 Validation experiment. Input image (top), strand segmentation (center), histogram of the strand orientation (bottom). In blue, the histogram as computed by the software; in red, the expected result.

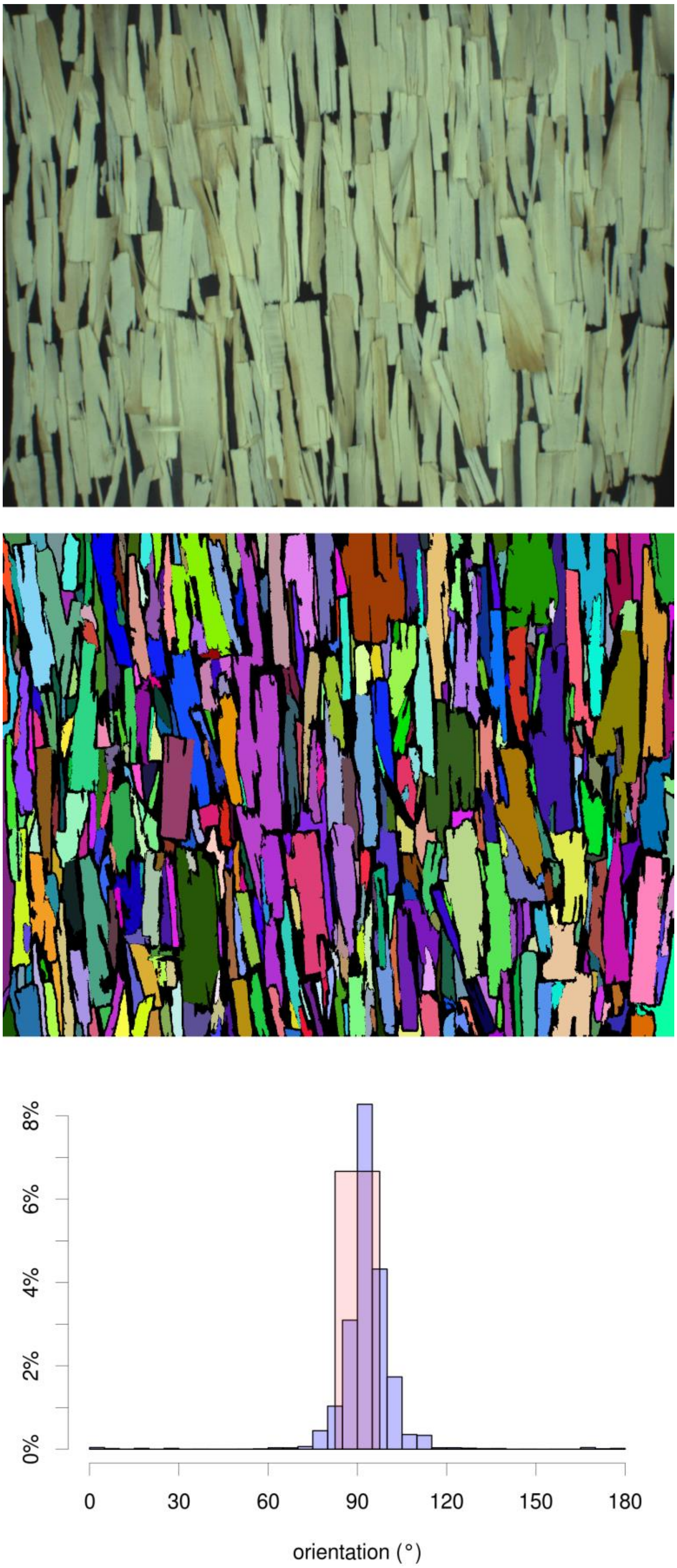

Figure 18 Validation experiment. Input image (top), strand segmentation (center), histogram of the strand orientation (bottom). In blue, the histogram as computed by the software; in red, the expected result.

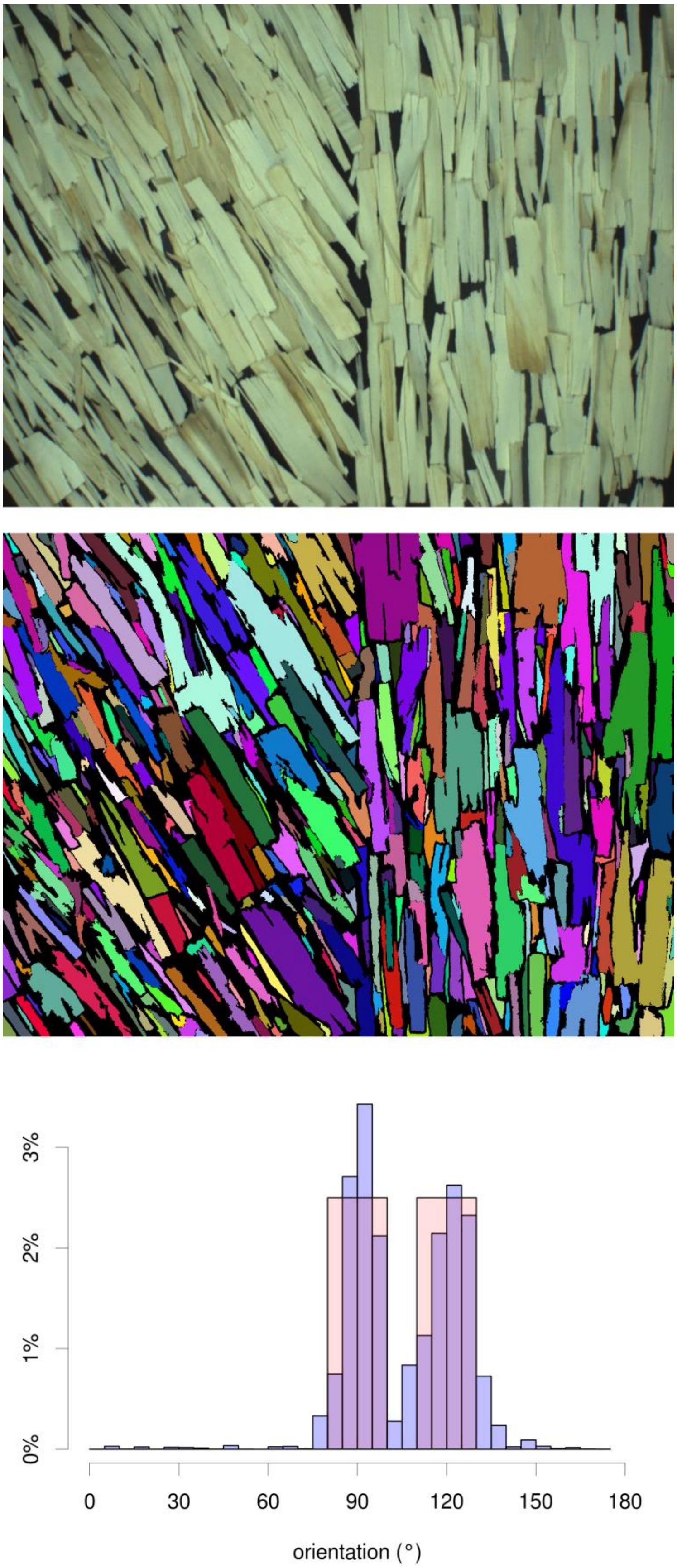

Figure 19 Validation experiment. Input image (top), strand segmentation (center), histogram of the strand orientation (bottom). In blue, the histogram as computed by the software; in red, the expected result.

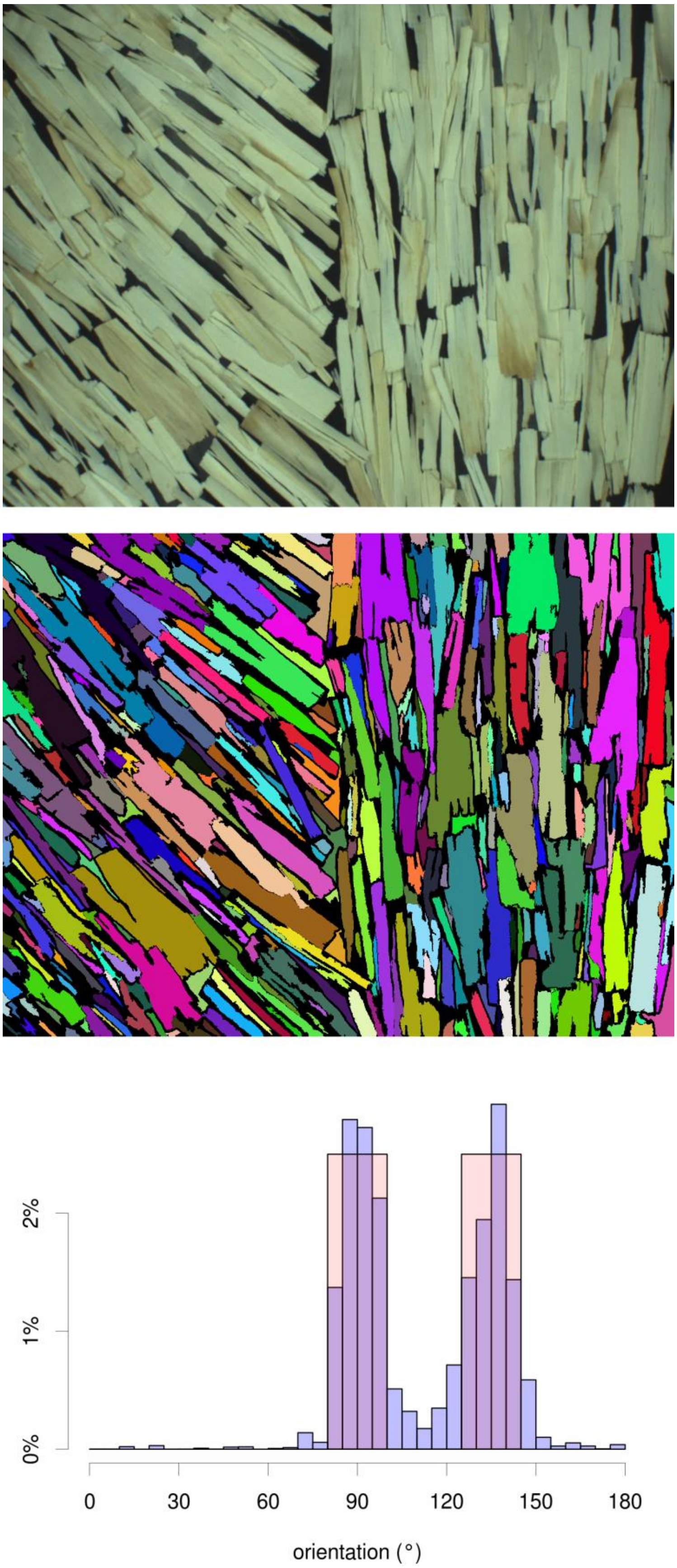

Figure 20 Validation experiment. Input image (top), strand segmentation (center), histogram of the strand orientation (bottom). In blue, the histogram as computed by the software; in red, the expected result.

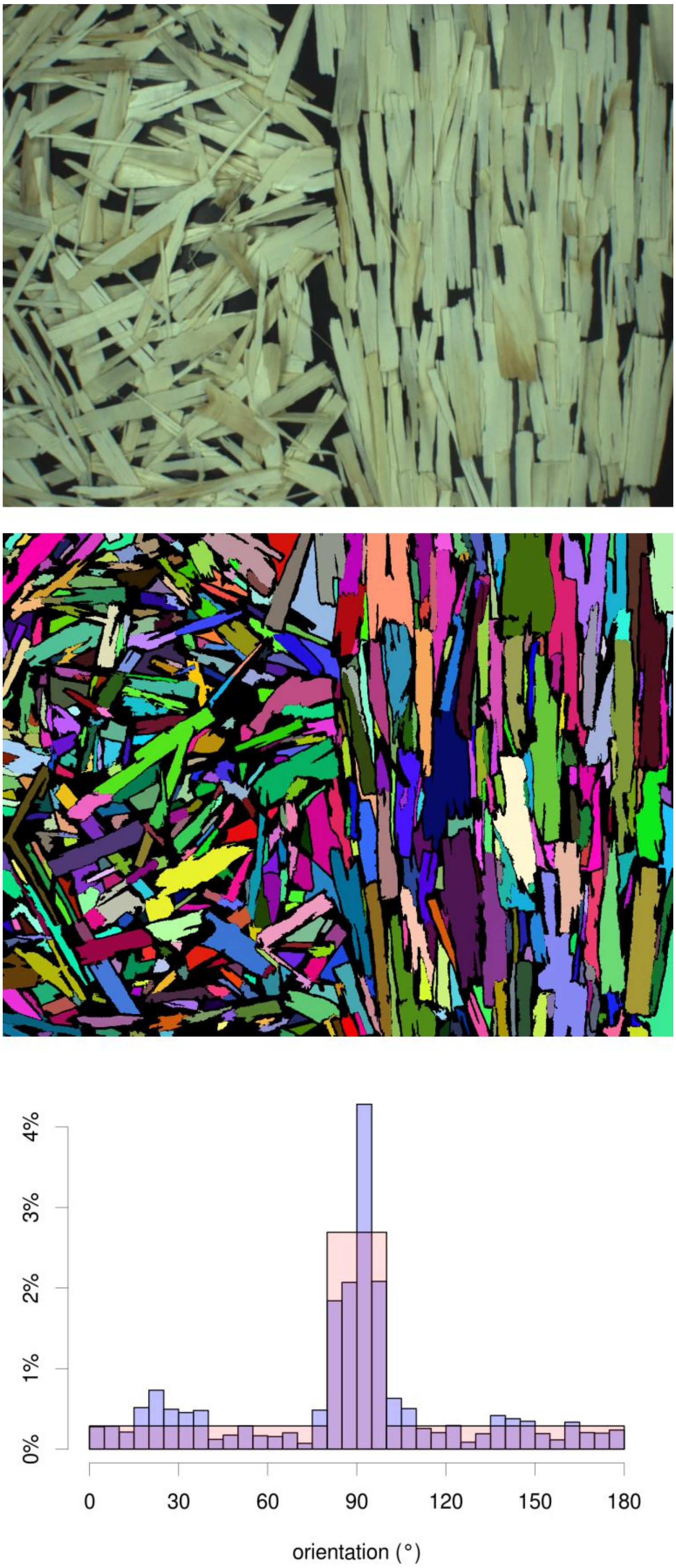

Figure 21 Validation experiment. Input image (top), strand segmentation (center), histogram of the strand orientation (bottom). In blue, the histogram as computed by the software; in red, the expected result.

Overall, the segmentation is visually accurate. There are sporadic mistakes: rarely different strands are being joined together or a strand is split in multiple parts, usually along the minor axis. This does not affect the final result, and indeed the comparison between computed and expected histograms show the effect on those mistakes is hardly apparent. The orientation detected by the software is remarkably consistent with the expectation.

The running time of the algorithm is also an important factor, as the software is intended for online monitoring. The experiment have been run on a regular computer (Intel Core i7-3770 processor). The processing took between 8 and 9 seconds to complete. The running time is almost entirely due to the clustering step. Indeed, mean-shift clustering is notoriously time consuming, due to the fact that the mean-shift procedure is run on every pixel of the image. Still, the current software already meets the requirements of the production system. Moreover, mean-shift is easily parallelizable and can be run on graphical processing units (GPUs). There are multiple such implementations available in well-known libraries such as OpenCV (Bradski 2000). In particular, GPU-implementations of mean-shift segmentation claim 50-100x speedups with respect to CPU (Li and Xiao 2009), bringing the running time well below half a second.

5 CONCLUSIONS

In this deliverable we have presented the image processing technique used to measure the orientation of wooden strands from single, one camera images. This is the simplest acquisition setup under consideration, which is an advantage in terms of costs and easiness of deployment. Our method works by locating each strand inside the image and computing the orientation from its contour. The first task required solving a difficult image segmentation problem, which we did by formulating the problem as a clustering one and designing the appropriate technique to solve it.

The software has been tested on laboratory images provided by UNIMI, simulating the imaging condition at the factory. The test images were designed so that the orientation distribution was known in advance and thus we had a baseline to compare against the result of the software. The experimental validation study shows the method is accurate in the segmentation and the estimated orientation distribution is very consistent with the expectation.

The goal of implementing a vision system and implementing a software method for measuring the orientation of the wooden strands on the conveyor belt can be considered completed. A possible improvement of the software would be to improve the processing time. Nevertheless, this is not critical requirement and it can be accomplished with just a minor effort.

6 BIBLIOGRAPHY

Arbelaez, P., M. Maire, C. Fowlkes, and J. Malik. 2011. "Contour Detection and Hierarchical Image Segmentation." *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 33 (5): 898–916. doi:10.1109/TPAMI.2010.161.

Bishop, Christopher M. 2006. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc.

Bradski, G. 2000. *Dr. Dobb's Journal of Software Tools*.

Canny, John. 1986. "A Computational Approach to Edge Detection." *Pattern Analysis and Machine Intelligence, IEEE Transactions on* PAMI-8 (6): 679–98. doi:10.1109/TPAMI.1986.4767851.

Comaniciu, Dorin, and Peter Meer. 2002. "Mean Shift: A Robust Approach Toward Feature Space Analysis." *IEEE Trans. Pattern Anal. Mach. Intell.* 24 (5). Washington, DC, USA: IEEE Computer Society: 603–19. doi:10.1109/34.1000236.

Estivill-Castro, Vladimir. 2002. "Why so Many Clustering Algorithms: A Position Paper." *SIGKDD Explor. Newsl.* 4 (1). New York, NY, USA: ACM: 65–75. doi:10.1145/568574.568575.

Fabio Scotti, Luis Magdalena. *Internal Report: Strand Distribution Analysis*.

Forsyth, David A., and Jean Ponce. 2002. *Computer Vision: A Modern Approach*. Prentice Hall Professional Technical Reference.

Fukunaga, K., and L. Hostetler. 1975. "The Estimation of the Gradient of a Density Function, with Applications in Pattern Recognition." *Information Theory, IEEE Transactions on* 21 (1): 32–40. doi:10.1109/TIT.1975.1055330.

Hastie, Trevor, Robert Tibshirani, and Jerome Friedman. 2001. *The Elements of Statistical Learning*. Springer Series in Statistics. New York, NY, USA: Springer New York Inc.

Kriegel, Hans-Peter, Peer Kröger, Jörg Sander, and Arthur Zimek. 2011. "Density-Based Clustering." *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 1 (3). John Wiley & Sons, Inc.: 231–40. doi:10.1002/widm.30.

Li, Peihua, and Lijuan Xiao. 2009. "Mean Shift Parallel Tracking on GPU." In *Pattern Recognition and Image Analysis*, edited by Helder Araujo, AnaMaria Mendonça, ArmandoJ. Pinho, and MaríaInés Torres, 5524:120–27. Lecture Notes in Computer Science. Springer Berlin Heidelberg. doi:10.1007/978-3-642-02172-5_17.

Lloyd, S. 1982. "Least Squares Quantization in PCM." *Information Theory, IEEE Transactions on* 28 (2): 129–37. doi:10.1109/TIT.1982.1056489.

Nishimura, T, and M P Ansell. 2002. "Fast Fourier Transform and Filtered Image Analyses of Fiber Orientation in OSB." *Wood Science and Technology* 36 (4): 287–307. doi:10.1007/s002260100114.