Comparative study of laser stripe detection algorithms for embedded real-time suitability in an industrial quality control context

Seher Colak¹,², Virginie Fresse¹,³, Olivier Alata¹, Thomas Gautrais¹ and Emmanuel Dumas²

¹ Hubert-Curien laboratory, UMR CNRS 5516, Jean-Monnet University of Lyon, France
² Pattyn Bakery Division, ZA de Marcouly, Maleville, France
³ E-mail: virginie.fresse@univ-st-etienne.fr

Abstract. Laser triangulation is often used for industrial quality control. In this context, the main issues are temporal performance, embedded suitability and precision. Laser line detection is the key step in laser triangulation. Several laser line detection algorithms were developed on the last decades. The object of this paper is to study the suitability of these algorithms towards the cited problematics. To reach this goal, we updated the algorithms precision comparison with some recent algorithms, we analyzed the execution time of these algorithms, and we studied the complexity of the algorithms to judge their adequacy toward embedded constraints. Analysis shows that none of the algorithms are perfect, so the choice is a compromise between precision, computation resources needs and execution time.

1. Introduction
Laser triangulation is a three-dimensional acquisition method often used for industrial quality control. An introduction to this method, also known as active range finding, can be found in [1] and [2]. Laser triangulation is specifically used in the food industry where contact with the product should be limited. In an industrial context, the main needs are: (1) a high temporal performance due to a fast continuous process and a real-time constraint, (2) a limited use of resources to respect embedded constraints, and (3) a precision constraint depending on the product for the quality control aspect.

Laser line detection is the key step to fulfil those needs. It consists in estimating the position of the laser line on each column of the acquired image at sub-pixel accuracy. The laser line detection processing time impacts time-related performances, its complexity impacts the use of resources and its precision impacts the precision of the measured height (with the calibration). Laser line detection is subject to specific noise sources. The two most important noise sources on laser line detection are (1) the Speckle noise and (2) the saturation noise. The Speckle noise is a multiplicative noise due to the coherent nature of the laser light. On the acquired laser line image, an important variation of intensity for pixels forming the laser line is visible. This leads to difficulties to find the peak of the laser line. The saturation noise is a non-linear noise due to a too long exposition of the image sensor to light. It leads to a difficulty to detect the peak of the laser line.

Several laser line detection algorithms were developed on the last decades, from different approaches, with different features and adapted to different contexts. The object of this paper is to study the suitability of these algorithms with the industrial quality control context. The authors of [3],...
[4] and [5] already made a comparison of laser line detection algorithms, but focused only on the precision. The main contribution of this paper is to update the algorithm comparison with some new algorithms, and to add an analysis of execution time and complexity to study the algorithms adequacy toward embedded real-time constraints. First, the chosen algorithms and the comparison methodology will be presented in section 2. Then, the algorithms will be compared according to their precision in section 3, their complexity in section 4, and their execution times in section 5. Section 6 concludes this paper.

2. Comparison platform
To compare the algorithms, first, we acquired 20 laser line images of an object of known dimensions. The object we used is a 850 mm long piece of metal composed of four pyramids with 12 mm vertical steps and 15 mm horizontal steps. Dimensions of the object are depicted on figure 1. This object is traditionally used to calibrate the laser triangulation system. A red filter has been put on the camera lens to avoid noise due to ambient light. The images have a 2048x1088 pixels resolution and were acquired with different exposure times. In images with a small exposure time, the laser line is subject to Speckle noise. On the contrary, in images with a high exposure time, the laser line is saturated. Images with average exposure times are classic images of laser triangulation in our context. Figure 2 to Figure 4 represent a section of the acquired laser lines images for different exposure times.

![Figure 1. Dimensions of the used object in mm.](image)

![Figure 2. Section of laser line acquired with a low exposure time (100 µs). Speckle noise is visible on the laser line.](image)

![Figure 3. Section of laser line acquired with an average exposure time (200 µs). The laser line is slightly saturated.](image)

![Figure 4. Section of laser line acquired with a high exposure time (400 µs). The whole laser line is saturated.](image)

A software platform has been created to compare the detection algorithms. The platform is presented on Figure 5. It works with two steps. In the first step, several laser line detection algorithms developed on C++ are applied to the images in order to have the laser line positions on the images. These algorithms are based on different principles: thresholding, approximation, interpolation, center of mass, FIR filtering or statistics. Based on previous studies, we did not consider approximation and interpolation methods which have a poor precision and execution time. Finally, we considered the following algorithms:

- Max: returns the position of the first maximum intensity pixel.
- Moy: Computes the average position of pixels whose intensity is maximum.
- Med: computes the median position of pixels whose intensity is maximum.
- CM: Obtains the position of the laser line at a sub-pixel accuracy, by computing the center of mass. Pixels around the maximum intensity pixel are used. The positions are weighted by the intensities of the pixels.
- BR: FIR filtering of the laser stripe profile by a [1 ... 1 0 -1 ... -1] filter to obtain the derivative of the profile. Then computes the 0 crossing point of the filtered signal.
• GA [3]: Gaussian approximation, assumes that the laser profile has a Gaussian shape and computes the average of this Gaussian function.
• GD [5], [10] and [11]: FIR filtering of the laser stripe profile by a Gaussian derivative to derivate and smooth the profile. Then, computes the 0 crossing point.
• MS [12]: Designed for lasers with uniform distribution. Computes the statistical moments of the pixels around the laser line. Then computes the position of the edges of the laser line and the middle position of the laser line.

Figure 5. Software evaluation platform for the comparison of the algorithms.

In the second step, statistical metrics are computed. These metrics are: standard deviation, bias, root mean square error, and computation time. These metrics enable to estimate the precision of the algorithms and their execution times.

3. Precision of the algorithms
The precision of the algorithms is typically estimated with the standard deviation. In this study, we added the bias and the root mean square error. To estimate these metrics, the software platform presented above is used with 20 images corresponding to different exposure times.

These metrics were computed for each height steps of the object. When it is not possible to obtain the position of the laser line (for example in case of occlusions), the result of the laser stripe detection is set to 0 and the result is not taken into account for the computation of the metrics. The obtained metrics for each height step are then averaged taking into account the number of valid samples.

3.1. Standard deviation
Standard deviation results are shown on Figure 6. The standard deviation tends to vary according to the exposure time. Some exposure time ranges enable a better standard deviation for some algorithms. Aside from the Max algorithm, all algorithms have a standard deviation between 0.2 and 0.6 mm. In average, the most precise algorithms are GD, CM, and MS.

For small exposure times (between 50 and 200 µs), the most precise algorithms are GD, BR and GA. This is coherent because these algorithms were designed for laser stripes with a Gaussian profile. For the highest exposure times (above 700 µs), the most precise algorithms are GD, CM, and Moy. For average exposure times (between 200 and 700 µs) the most precise algorithms are GD, CM and MS. For the Max algorithm, the more the laser line is saturated, the more, the precision is low. This is due to the fact that the laser line is wider on the images, so the returned position is not the center of the laser line, but its edge.
3.2. Bias

Bias results are shown on Figure 7. Algorithms Med, Moy, CM, BR, GD, and GA have a bias between -0.5 mm and +0.5 mm. The bias tends to decrease with the exposure time. The bias is lower for average exposure times. Algorithms with the lowest bias are GD, GA and Moy. For the Max algorithm, the same phenomenon as for the standard deviation happens.

We could also see that the bias is the smallest for the 250 µs exposure time. This is due to the fact that the system calibration was done with images at this exposure time. A high bias is not a problem because it can be corrected by the calibration.

3.3. Root mean square error

Root mean square error gathers the bias and the standard deviation. To compute the root mean square error, we first compute the mean square error for each height steps. Then, the mean square error is averaged while taking into account the number of valid samples. Finally, a square root has been applied to obtain the root mean square error. The results for this metric are shown in Figure 8.

As the bias is smaller for average values of exposure time, the root mean square error is lower for this exposure time range. For smaller and higher exposure times, the error is higher but always between 0.2 and 0.7 mm for six of the algorithms. This is mostly due to the bias and the fact that the standard deviation is higher for small and high exposure times. The only algorithms with a high error rate are Max and MS due to their high bias.

This metric shows more variability on the precision depending on exposure time. For high exposure times (above 700 µs), the best algorithms are the Moy-Med, the GD and the MS. Between
400 and 700 µs, the best algorithms are GD, CM, and Moy-Med. Between 100 and 400 µs, the best algorithms are GD, CM, and BR. Finally, under 100 µs, the best algorithms are GD, BR and GA.

![Figure 8. Root mean square error as a function of exposure time.](image)

3.4. Discussion
A summary of the precision metrics is shown on Table 1. We ranked the algorithms according to their average precision on each of the three metrics. The smallest number corresponds to the most precise algorithm. The most precise algorithms are the GD and the CM. The Max algorithm is the least precise. The other algorithms seem to have an equivalent precision between each other. Some of them, like the Moy, have a better accuracy on saturated laser lines. Others, like the BR, have a better accuracy on narrow and noisy laser lines.

| Algorithm | Max | Med | Moy | CM | BR | GD | MS | GA |
|-----------|-----|-----|-----|----|----|----|----|----|
| Std. dev. | 5   | 4   | 4   | 2  | 4  | 1  | 3  | 4  |
| Bias      | 5   | 2   | 2   | 3  | 3  | 1  | 4  | 2  |
| RMSE      | 5   | 3   | 3   | 2  | 3  | 1  | 4  | 3  |

4. Complexity of the algorithms
Studying the complexity of the algorithms allows to decide which kind of computing unit is the best suited. For example, an application needing a high number of simple operations can be implemented on a massively parallel architecture like an FPGA or a GPU, while an application needing a few number of complex operations is better on a CPU. The complexity of the algorithms can be studied by an estimation of the number of operations, their type and their estimated memory consumption. Here, the memory consumption is not studied.

4.1. Type of operations
Operations can be classified in three classes for an embedded implementation. The first class contains most of the basic operators like addition, subtraction, multiplication, and comparisons. The second class contain operators that can be obtained by simple algorithms like division and square root. The final class contain operators that are obtained by complex algorithms like logarithm, exponential, and other trigonometric operators. Examples of algorithms to obtain these operators can be found in [13].

Algorithms can be classified according to the type of operators they need. This classification is shown on Table 2. The algorithms Max and Med, only need first class operators. Algorithms Moy, CM, BR, GD and MS need first and second class operators, and the algorithm GA need first, second and third class operators.
Table 2. Class of the algorithms operators.

| Algorithms | Max | Med | Moy | CM | BR | GD | MS | GA |
|------------|-----|-----|-----|----|----|----|----|----|
| Operators class | 1   | 1   | 1, 2 | 1, 2 | 1, 2 | 1, 2 | 1, 2 | 1, 2, 3 |

4.2. Number of operations

The number of operations for each algorithm has been estimated. From these estimations we can deduce the algorithms complexity. For these estimations, we will note $N$ the number of lines on the image, and $L$ the number of pixels used to compute the position of the laser line. The complexity of the algorithms according to these variables is shown on Table 3. The Max algorithm has a linear complexity to the variable $N$. Moy, Med, CM, GA, and MS have a linear complexity to the variables $N$ and $L$. The algorithms BR and GD have an $N \times L$ complexity. So, these two algorithms are the ones that will need more operators when $N$ and $L$ will increase.

Table 3. Complexity of the algorithms to variables $N$ (number of lines on the image) and $L$ (number of pixels used for computation).

| Algorithms | Max    | Med    | Moy   | CM    | BR    | GD    | MS    | GA    |
|------------|--------|--------|-------|-------|-------|-------|-------|-------|
| Complexity | Linear $N$ | Linear $N$ | Linear $N$ | Linear $N$ | $N \times L$ | $N \times L$ | Linear $N$, $L$ | Linear $N$, $L$ |

5. Execution time of the algorithms

The execution times of the algorithms were measured on an Intel Core i7 CPU (2.20 GHz) through the software platform. This study contains high limitations because it has been done only on one CPU. The execution times were measured for two versions of the algorithms. In the first version, the whole image is processed. In the second version, pixels with an intensity under a defined threshold were removed. The time used to remove those pixels was added to the time used to compute the laser line position.

![Figure 9. Execution time of the algorithms.](image)

The results are available on Figure 9. For both versions of the algorithms, the Max algorithm is the faster, while GD and BR are the slowest. The other algorithms (CM, AG, Med, MS and Moy) have an average execution time. Removing non-significant pixels improve the execution time. Algorithms with a threshold process are 2.5 times faster than the algorithms which process the whole image.

For the version with threshold, the average execution time on CPU is approximately 19 ms. With this execution time, only 52 images can be processed in a second which is not enough in a continuous
production context. Most of the image sensors dedicated to industry can acquire more than 300 images per second.

6. Conclusions and perspectives

Several laser line detection algorithms have been studied in this paper. The properties of the algorithms have been analyzed and compared. None of these algorithms are perfect, so a choice for a given application must be a compromise between precision, computing resource requirements and execution time.

According to our study, GD is the most precise algorithm regardless of the exposure time. Its disadvantage is its complexity, and especially its number of operations which increases rapidly. Its second disadvantage is its CPU execution time that can be increased by replacing the computing unit with a DSP, an FPGA or a GPU. The CM algorithm is a compromise between complexity and precision. These two algorithms are currently the most used algorithms for laser line detection. The results for the MS algorithm are disappointing, especially for the high exposure times. Indeed, this algorithm was developed for large saturated laser lines but has a low precision in these cases.

This study has limitations, especially for the execution time and bias results. Since the bias depends on the calibration, it is difficult to do a bias study without taking into account the calibration procedure. For the execution time, as it depends on the architecture of the CPU and its frequency, it is difficult to have a reliable study with a single CPU. This study on CPU is a first step before another analysis on FPGA or other architectures.

References

[1] Sanz J L C 1988 Advances in Machine Vision Sanz J L C (Springer-Verlag)
[2] Szeliski R 2010 Computer Vision: Algorithms and Applications (Springer)
[3] Fisher R B and Naidu D K 1996 A comparison of algorithms for sub-pixel peak detection, Image Technology. Sanz J.L.C. (eds) (Berlin, Heidelberg: Springer) 385–404
[4] Haug K and Pritschow G 1998, Robust laser-stripe sensor for automated weld-seam-tracking in the shipbuilding industry Proc. of the 24th Annual Conf. of the IEEE Industrial Electronics Society, 1998. IECON 98 2 pp1236–41
[5] Forest J, Salvi J, Cabruja E and Pous C 2004 Laser stripe peak detector for 3d scanners. A FIR filter approach Proc. of the 17th Int. Conf. on Pattern Recognition, 2004. ICPR 2004 3 pp646–649
[6] Barreto S V F, Sant’Anna R E and Feitosa M A F 2013 A method for image processing and distance measuring based on laser distance triangulation IEEE 20th Int. Conf. on Electronics, Circuits, and Systems (ICECS 2013) pp695–698
[7] Usamentiaga R, Molleda J and Garcia D F 2010 Fast and robust laser stripe extraction for 3D reconstruction in industrial environments Machine Vision and Applications 23 179–112
[8] Prats M, Fernandez J J and Sanz P J 2012 Combining template tracking and laser peak detection for 3D reconstruction and grasping in underwater environments IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS) pp106–112
[9] Blais F and Rioux M 1986 Real-time numerical peak detector, Signal Processing 11 145–155
[10] Steger C 2013 Unbiased extraction of lines with parabolic and Gaussian profiles Computer Vision and Image Understanding 117 97–112
[11] Li Q, Zhang Y, Zhang X, Wang S and Xie F 2013 Statistical behavior analysis and precision optimization for the laser stripe center detector based on Steger’s algorithm Optics Express 21 13442–449
[12] Sun Q, Chen J and Li C 2015 A robust method to extract a laser stripe centre based on grey level moment Optics and Lasers in Engineering 67 122–127
[13] Muller J M 2006 Elementary Functions: Algorithms and Implementation (Boston: Birkhäuser)