Image search, Match the fastener

Raphaël Bulle, François Der Hovsepian, Francisco Nicolás, Jean-Christophe Sick

Image search, Match the fastener

Raphaël Bulle[1], François Der Hovsepian[2], Francisco Nicolás[2], and Jean-Christophe Sick[3]

[1]University of Luxembourg, Esch-sur-Alzette, Luxembourg
[2]Université de Strasbourg, Strasbourg, France
[3]RAYCE EURL (ARaymond Network), Saint-Louis, France

Introduction

This work has been initiated during the Maths-Companies study group (Semaine d'Étude Maths-Entreprise, SEME) organised by AMIES (Agence pour les Mathématiques en Interaction avec l'Entreprise et la Société), CNRS (Centre National de la Recherche Scientifique) and the University of Strasbourg from 12th to 18th of November 2018. During this week we have decided to work on the subject given by Rayce Eurl, the ARaymond network centre of expertise, represented by Jean-Christophe Sick innovation project manager at Rayce.

We would like to thank all the organisers of this study group week as well as Jean-Christophe for his availability during this week, all our fructuous discussions have allowed us to benefit from his industrial perspective. We also thank Mr. Zakaria Belhachmi who has been our academic supervisor, his extensive knowledge on fields related to the subject was a precious help. Finally we thank Céline Caldini-Queiros and Myriam Maumy-Bertrand for their support during this week.

Subject description proposed during the SEME

In an assembly situation (male - female parts interaction), from a real "female" part, by using photo search innovative technologies, propose a selection of matching "male" parts contained in the products catalogue. Real and virtual assembly parts such as photos from the products catalogue of ARaymond and possibly 3D models

will be provided. To start, typical male parts are a selection of clips and female parts are plates with holes that can receive the clip.

Our reformulation of the problem

We reformulate the problem statement as follow:

From one or several pictures of a "female" part constituted of a plate with a single hole in it, identify the shape and characteristic dimensions of this hole and find matching clips in ARaymond's database of "male" parts.

We can split the problem in two main parts:

1. process the picture(s) of the "female" part to extract the hole dimensions,
2. propose "male" parts in the ARaymond database that best match the hole.

It is important to notice that, unlike "male" parts, "female" parts are not manufactured by ARaymond but manufactured by their customers. As a consequence, the ARaymond company has at its disposal (almost) every information concerning the "male" parts: geometrical parameters such as length, diameter... but also physical parameters such as material, elasticity... but has no information about the "female" parts. The goal is to propose the most accurate solutions to their customers with the minimum requirement of informations about their products. The idea being to simplify the process: by using only one or a few pictures of the "female" parts sent by the customer, ARaymond would like to be able to propose a selection of matching clips.

Since all information about "male" parts are stored in a digital database of ARaymond, the second part of the above problem is reduced to a database search. Thus, the main challenge lies in the first part of the problem: process the picture(s) and extract the dimensions of the "female" part hole. This is why we have chosen to focus on this part of the problem.

In the following we will only focus on the geometrical aspects of the problem, neglecting the mechanical aspects of physical constraints that occur when we insert the clip. Consequently, we will say that a male and a female parts match when their corresponding dimensions are similar enough.

We can also split this first part into smaller steps:

1. process the picture(s) to detect edges (e.g. entrance and exit of the hole) and extract useful information,

2. measure the dimensions of the hole (e.g. diameter for a circle, edge length for a square, depth...).

A first approach based on projective geometry

Our early thoughts of the problem reminded us of 3D reconstruction from multiple images and 3D-2D registration problems, which brought back epipolar geometry memories. This approach allows to extract geometrical information from two or more pictures and has been widely described, for example in [5] which explains the process using two, three or more images. The methods based on these concepts can achieve high accuracy and are used in computer vision and 3D scanning technologies. However, once we had more details on the industrial context, we decided to drop this approach entirely due to its intrinsic requirements, and to switch to a simpler one, using only one picture.

1 Our final approach: a single picture

Several reasons lead us to simplify the process. Using the inherent constraints of the framework and adding constraints on the shooting process we aimed at reducing both the computational complexity of the algorithm and the total time of the process.

The first reason was that female parts come from industrial components and, consequently, have standard shapes. For example most of their holes are circular and drilled orthogonally to the top side of the plate. This reduces the list of shapes we have to detect and eases image processing.

A second reason was the fact that we only need a few parameters of the female part, like radius and depth in case of circular hole, to browse a matching male part. So the (costly) reconstruction of the full 3D object would provide a lot of unnecessary information.

The third reason was the fact that the shooting process would be carried out by ARaymond's industrial customers, consequently they have an interest in making it as simple as possible.

The quality of the picture would clearly influence the accuracy of such an algorithm. Since our measurements in the picture are based on pixels, the accuracy is directly linked to the picture resolution. However, we assumed today's cameras were good enough for us not to worry about any requirement on this side, at least for our proof of concept.

1.1 Framework and constraints on the shooting process

We limit ourselves to the case of a **circular** hole, **drilled orthogonally** to the top side of the plate and with **small depth** (i.e. at most of the same order of magnitude as the diameter). Once again these two last assumptions are realistic from an industrial point of view. Then, we need to measure only two parameters: the **diameter** and the **depth**. We also assumed, in order to simplify the process for the customer, that he would have to take **only one picture** of the female component.

Figure 1: Example of female components satisfying our constraints.

Even with these constraints the problem still remains complicated due to all the possibilities in shooting. A lot of parameters to describe the position of the camera are introduced: the distance between the camera and the region of interest or the different angles of shooting.

All our measurements have to be made from a single picture, so we should reduce the number of parameters that could induce deformations of the object on the picture

as much as possible. Also, having a reference size on the picture will be necessary. This reference size can be, for example, a circular sticker placed right next to the hole of the female component. This reference sticker would be such that:

1. it is flat (e.g. like a piece of paper),
2. all its dimensions are known (e.g. its diameter in the case of a circular sticker) and are of the same order of magnitude as the dimensions of the hole,
3. it is placed right next to the region of interest (i.e. right next to the hole),
4. it has no details on it (e.g. it is uniform in colour) and it is clearly recognisable on the picture.

Moreover we assume that the picture is centred between the sticker and the hole in order to reduce optical aberrations that could occur on the boundary of the picture.

The position of the camera can be characterised by four parameters: three angles α, β and γ and the distance d between the camera and the region of interest. These parameters are represented on figure 2. On this figure, the orange rectangle is the top side of the female component (the boundary may not be visible on the picture). The black and blue circles in the rectangle are respectively representing the entrance of the hole and a circular sticker placed next to it. Points H and S are the centres of the hole and of the sticker, the point O is where the picture should be centred (optimally O is the point on the line (HS) which is at equal distance from the two circles). The point P represents the camera. The triplet (α, β, d) are the spherical coordinates of the point P. In addition, we need a third angle, γ describing the rotation of the camera on itself around the axis joining the middle of the female part and the point P. The combination of these four parameters induces deformations of the object on the picture. The green vectors around P are the respective images of the orthogonal basis (in black) through the composition of the different rotations associated to angles α, β and γ and through the translation of vector $\overrightarrow{OP}$.

The idea is then to fix two of the three angles describing the position of the camera by taking,

$$\alpha \approx \frac{\pi}{2} \text{ and } \gamma \approx 0.$$

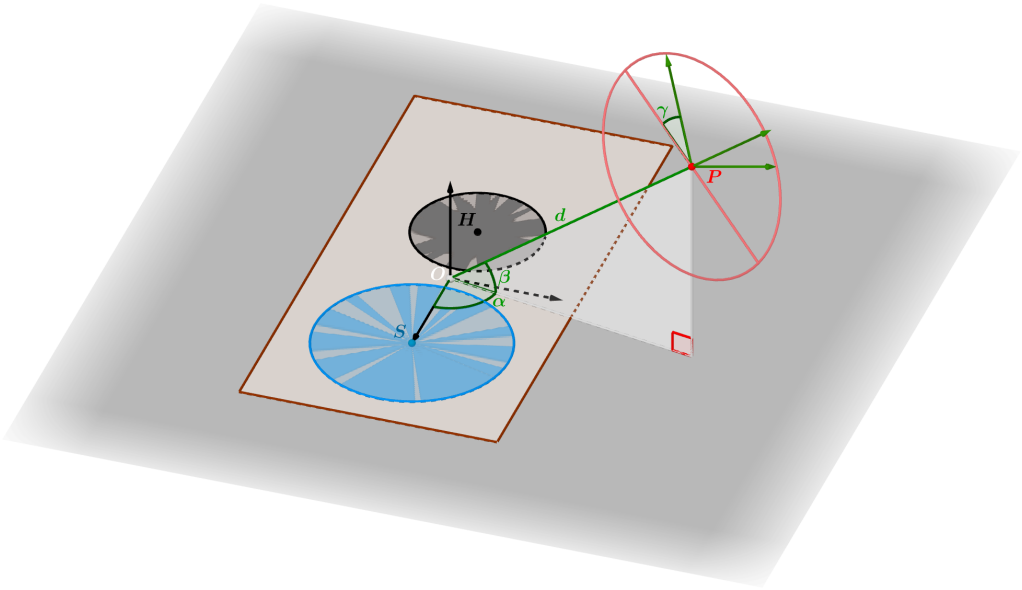

Figure 2: 3D representation of the shooting process.

In the next figure we show pictures which do not satisfy either the size reference or the shooting process constraints. In these pictures the sticker is the white disk of paper.

Figure 3: Examples of wrong pictures.

The last constraint will be on the angle β. If β is too small, then we will not be able to see the bottom of the hole, which makes the depth impossible to measure. Similarly, if β is too close to $\pi/2$ then the side of the hole will not be visible (or not visible enough) on the picture which - at best - would degrade the accuracy of measurement of the depth.

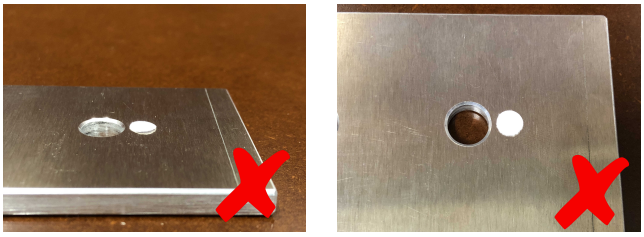

Figure 4: Examples of wrong pictures.

Finally, here is an example of a correct picture that satisfies all the constraints,

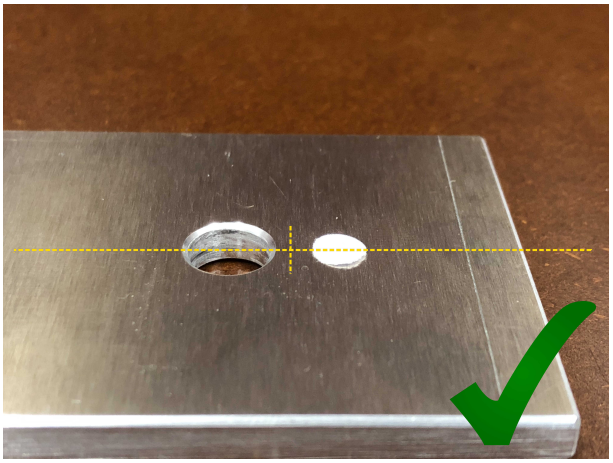

Figure 5: Example of good picture.

Here the horizontal yellow dashed line should be superimposed on the (HS) line of figure 2 and the cross in the middle of the picture should be on the point O. As we can see on this last picture, the shooting requirements make sense and can be easily satisfied.

We could also imagine a program which adds this yellow dashed line with the central cross on the screen of the camera to facilitate the shooting process.

We will see further in details how we deduce the real dimensions of the hole from this kind of picture. We can already see that the apparent diameter of the sticker on the picture along the horizontal yellow dashed line is only influenced by the distance

d and not by the angle β since the rotation of the camera is made around this axis.

The next challenge is to process this picture in order to make the edges of the hole and of the sticker visible by a computer.

1.2 Image processing

There is an extensive literature on this subject describing several different methods from noise reduction to corner detection, shape recognition, image segmentation or even texture recognition.

Here we have restricted ourselves to geometrical considerations and therefore did not look at the mechanical aspects of clip insertion. Moreover, since we have assumed the hole to be circular, we are more interested in edge detection than in shape recognition. Consequently we already know that, on the picture, the hole will have an elliptic shape due to the projection. However, in the general case when the shape of the hole is not known, there are still methods, for example [7], which allow to detect corners in a picture. Knowing the corners, we would then be able to deduce the shape of the hole.

But let us focus on edge detection. These methods are mainly based on the following procedure:

1. turn the picture into grayscale,
2. filter the noise,
3. study the brightness "flux" (gradient of the brightness) to detect edges or pseudo-edges,
4. distinguish the true edges from the fake ones,
5. process the edges to extract the wanted information.

A classical method of edge detection has been introduced by John Canny in 1986 in [2]. This is the method we are going to discuss in more details. Moreover, from a practical point of view, this method can easily be entirely implemented with a Python library called OpenCV [1] which is under a BSD license and hence free for both academic and commercial use. All the examples in this paper are computed

using Python and OpenCV.

The Canny edge detection is based on these four steps:

1. **Noise filtering**,
2. **Intensity gradient computation**,
3. **Non-maximum suppression**,
4. **Hysteresis thresholding**.

Before getting into these steps, let us define some useful tools. Let assume that we are working on a grayscale picture composed by a grid of $R \times C$ pixels. We can map this grid of pixels to a set of coordinates $E_{R,C} := \{0, \cdots, R-1\} \times \{0, \cdots, C-1\}$. In the following we will denote this space by E and write the subscripts only when they differ from R and C, also we will denote E_r the space $E_{r,r}$ for r an integer. We can define the two-dimensional discrete functions,

$$\begin{array}{rccc} f: & E & \longrightarrow & \mathbb{R} \\ & (x,y) & \longmapsto & f(x,y), \end{array}$$

which associate a certain real value to each pixel coordinates. Let us denote $\mathbb{R}^E$ the space of all the two-dimensional discrete functions. Then for any function f in $\mathbb{R}^E$ we can associate a matrix $A = (a_{i,j})_{(i,j)\in E}$ defined by,

$$a_{i,j} = f(i,j), \qquad \forall (i,j) \in E.$$

Moreover we will call *kernel* a function κ in $\mathbb{R}^{E_{2k+1}}$ where k is an integer. We define the convolution of a function f in $\mathbb{R}^E$ with the kernel κ, denoted $\kappa * f$, as a new function of $\mathbb{R}^E$ defined by,

$$\kappa * f(i,j) = \sum_{(h,l)\in E_{2k+1}} \kappa(h,l) f(i+1-h, j+1-l) \mathbb{1}_E(i+1-h, j+1-l), \quad \forall (i,j) \in E.$$

Example, for a kernel κ in $\mathbb{R}^{E_3}$ ($k = 1$) and a function f in $\mathbb{R}^{E_{3,4}}$ with respective matrices:

$$K = \begin{bmatrix} \kappa(0,0) & \kappa(0,1) & \kappa(0,2) \\ \kappa(1,0) & \kappa(1,1) & \kappa(1,2) \\ \kappa(2,0) & \kappa(2,1) & \kappa(2,2) \end{bmatrix} \text{ and, } A_f = \begin{bmatrix} f(0,0) & f(0,1) & f(0,2) & f(0,3) \\ f(1,0) & f(1,1) & f(1,2) & f(1,3) \\ f(2,0) & f(2,1) & f(2,2) & f(2,3) \\ f(3,0) & f(3,1) & f(3,2) & f(3,3) \end{bmatrix},$$

the convolution $\kappa * f(0,0)$ will be given by,

$$\kappa * f(0,0) = \kappa(0,0)f(1,1) + \kappa(0,1)f(1,0) + \kappa(1,0)f(0,1) + \kappa(1,1)f(0,0),$$

the convolution $\kappa * f(2,2)$ is given by,

$$\begin{aligned} \kappa * f(2,2) &= \kappa(0,0)f(3,3) + \kappa(0,1)f(3,2) + \kappa(0,2)f(3,1) + \kappa(1,0)f(2,3) \\ &+ \kappa(1,1)f(2,2) + \kappa(1,2)f(2,1) + \kappa(2,0)f(1,3) + \kappa(2,1)f(1,2) \\ &+ \kappa(2,2)f(1,1) \end{aligned}$$

Now let us get into each step of the Canny edge detection algorithm. We assume that f is a function of $\mathbb{R}^E$ which associates a grayscale image to an input picture, by giving the brightness of each pixel.

Noise filtering

This step aims to reduce the noise in the image in order to avoid the detection of fake edges. It consists in blurring the image with a Gaussian filter. A Gaussian filter of size $k \in \mathbb{N}$ is a kernel g in $\mathbb{R}^{E_{2k+1}}$ defined by,

$$g(i,j) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{(i-(k+1))^2 + (j-(k+1))^2}{2\sigma^2}\right), \quad \forall (i,j) \in E_{2k+1},$$

where the real σ^2 is the Gaussian filter's variance.

Then we blur the grayscale image using the convolution of the function f by the kernel g to create the blurred image. This blurred image can be associated to the function b in $\mathbb{R}^E$, defined by,

$$b := g * f.$$

The choices of the variance σ and the size k will affect the performance of the detector. For example, the larger the variance is the lower the detector's sensitivity to noise. In the same way, the error in edges detection will increase while we increase the Gaussian filter variance. The same can be said (to a lesser degree) for the size of the Gaussian filter. Here are some examples of Gaussian filters with different sizes and variances:

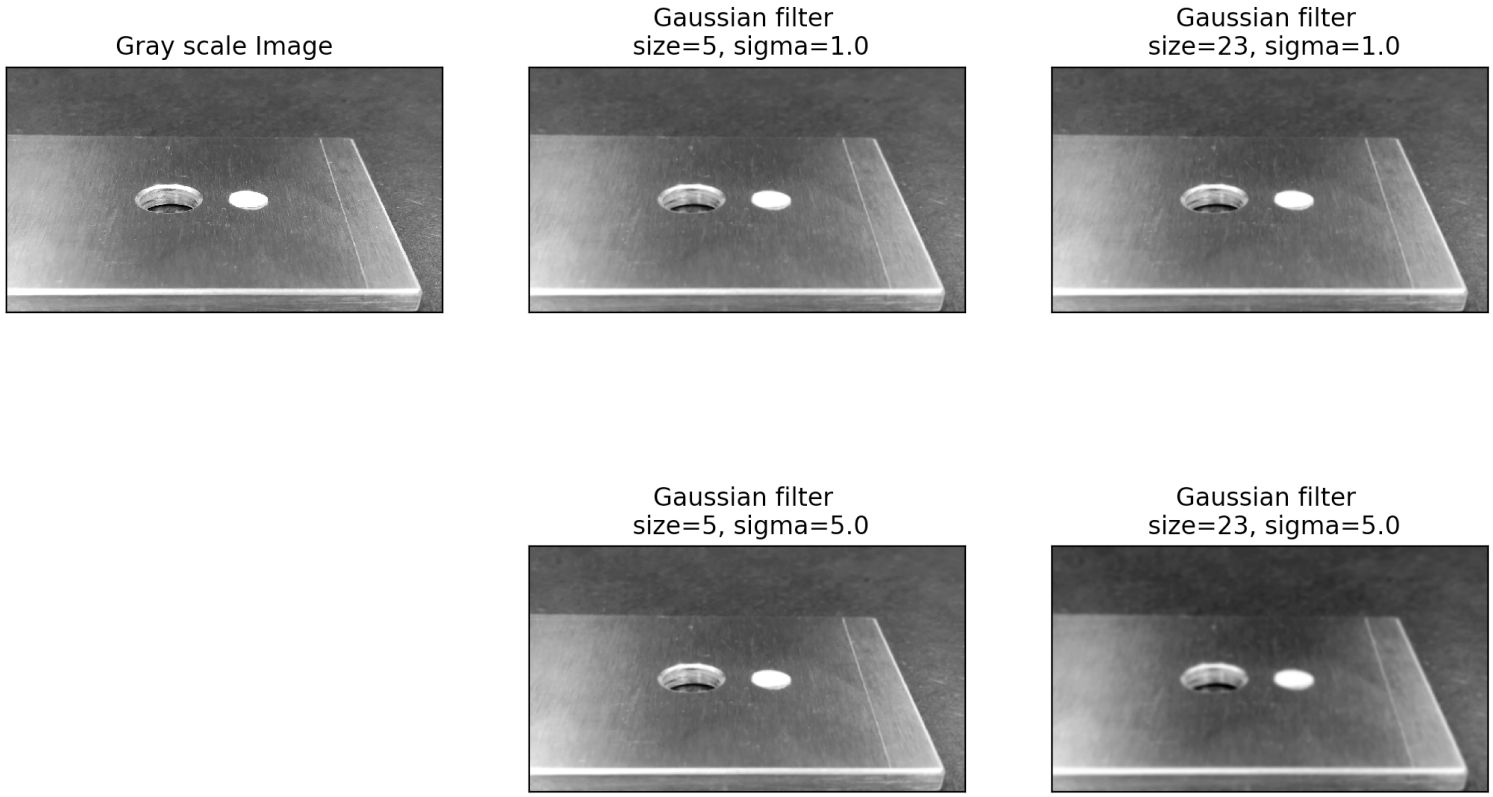

Figure 6: Examples of different Gaussian filters.

Of course, these parameters depend on the input image and have to be estimated from this image. Some solutions to this problem have been proposed such as in [3]. Also, Gaussian filters are not the only way of filtering the noise in an image. Techniques using anisotropic diffusion, as for example in [7], are more and more used.

Intensity gradient computation

A classical way to compute the intensity gradient in Canny edge detection is by using the Sobel operator (also called Sobel-Feldman operator) which as been introduced first in [4]. This operator is defined from the approximations of horizontal and vertical derivates respectively d_y and d_x which are kernels in $\mathbb{R}^{E_3}$ defined by their matrices,

$$d_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \text{ and } d_x = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}.$$

Then the derivatives of our blurred image are computed by convolution:

$$G_x = d_x * b \text{ and } G_y = d_y * b,$$

giving two functions G_x and G_y which can be seen as "discrete partial derivatives" (sometimes called *Sobel derivatives*) of b. Finally, the gradient magnitude is given by,

$$G(i,j) = \sqrt{G_x(i,j)^2 + G_y(i,j)^2}, \quad \forall (i,j) \in E,$$

and its orientation is given by,

$$\Theta(i,j) = \left[\text{atan2}\big(G_y(i,j), G_x(i,j)\big)\right], \quad \forall (i,j) \in E,$$

where $\left[\,\cdot\,\right]$ denotes a rounding operation which associates the output of the function atan2 to one of these eight angles,

$$\left\{0, \frac{\pi}{4}, \frac{\pi}{2}, \frac{3\pi}{4}, \pi, \frac{5\pi}{4}, \frac{3\pi}{2}, \frac{7\pi}{4}\right\},$$

depending on the position on the trigonometric circle as follow:

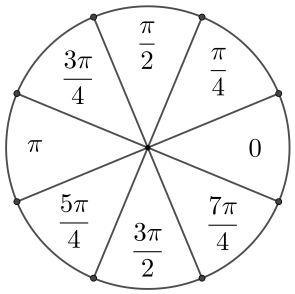

Figure 7: Rounded value for angles in each part of the circle.

There exist alternatives to the Sobel operator such as Roberts cross, Prewitt operator or Scharr operator. They use different kernels to compute the intensity gradient. Here are examples where we have applied the Sobel operator to compute the Sobel derivatives and the intensity gradient magnitude on our grayscale image and on a blurred example:

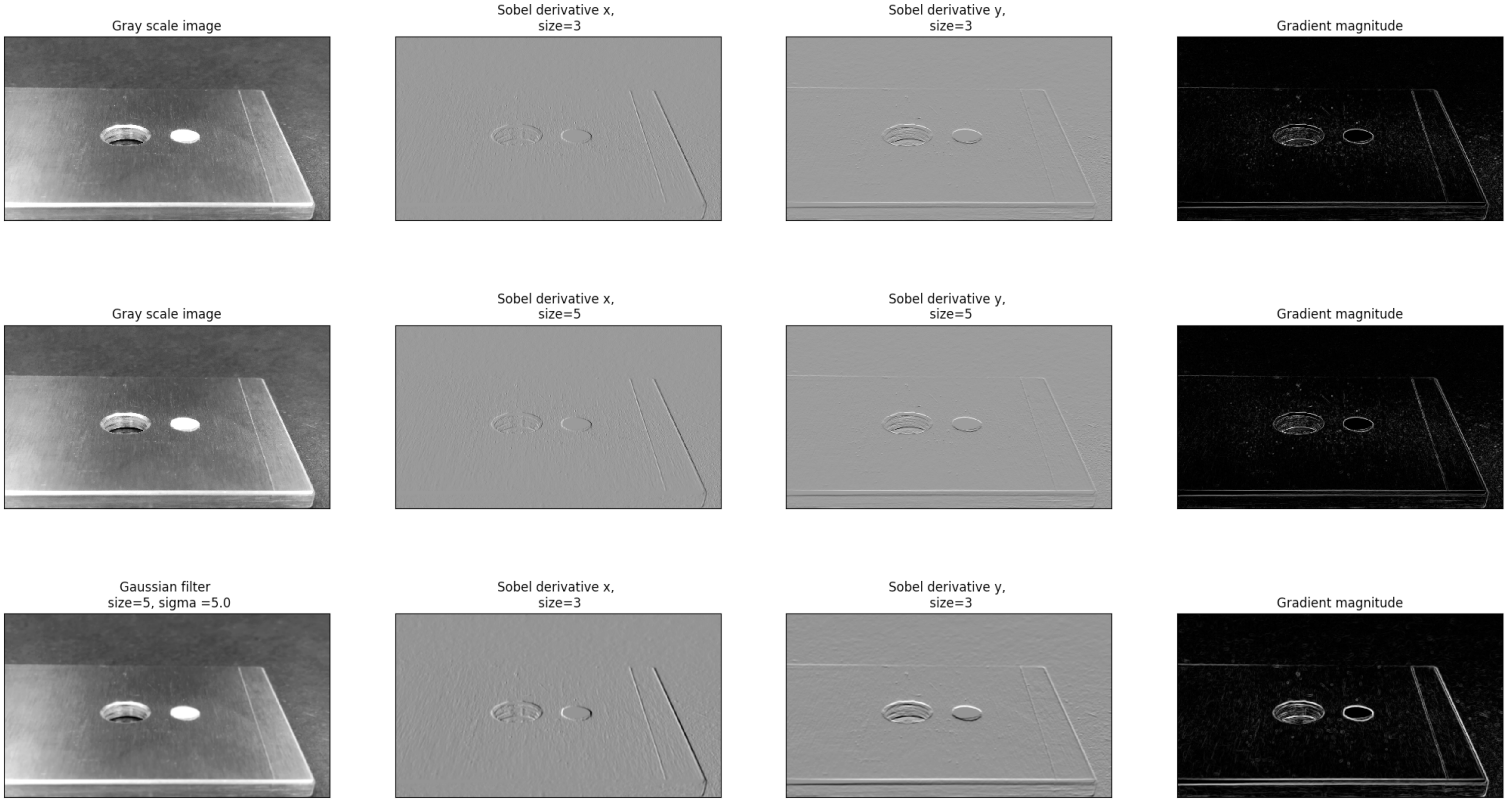

Figure 8: Examples of Sobel derivatives and gradient magnitude for the original picture and a blurred copy.

As we can see in figure 8, to change the size of the Sobel kernel does not have a great impact on the result but the choices in the Gaussian filter's parameters before computing the Sobel derivatives has an impact on the results.

Non-maximum suppression

The next step consists in the detection of local maxima in the magnitude of the gradient in order to find possible edges. The idea being that in an edge point the brightness intensity of the picture should highly change in the direction of the gradient. So at the end we will end up with a binary picture with, say white pixel at each local maxima of the gradient and black pixels elsewhere.

To do so we go through the image pixel after pixel, using the information on the gradient computed in the previous step to compare the gradient magnitude of the current pixel with those of its two neighbours in the direction of the gradient. If the gradient magnitude in the current pixel is greater than the magnitude of its neighbours, we "keep" this pixel, say we turn it into a white pixel. Otherwise we turn it into a black pixel.

In the next figure, the magnitude of the gradient is represented in shades of gray. In the case of pixel A, its gradient magnitude is higher than its two neighbours B and C in the gradient direction, so we should turn it on (e.g. to white) in the original picture.

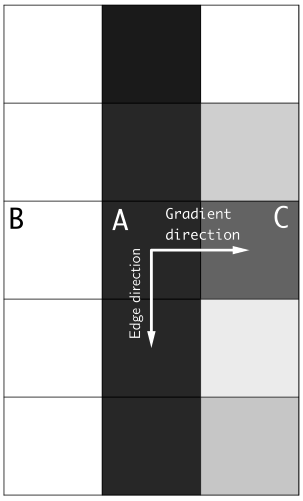

Figure 9: Non-maximum suppression procedure.

Hysteresis thresholding

This step follows directly the previous one in the loop over all pixels in the case where the pixel has been identified as a local maximum. In this case we still have to determine if this pixel really belongs to an edge or not. First we fix two thresholds $\alpha_{\max}$ and $\alpha_{\min}$. Then if the gradient magnitude at our pixel is greater than $\alpha_{\max}$ we keep it on as part of an edge. Otherwise, if the gradient magnitude is lower than $\alpha_{\min}$ we discard it. Finally, if the magnitude falls between $\alpha_{\max}$ and $\alpha_{\min}$ we decide to accept the pixel or not depending if it is connected to a pixel with magnitude over $\alpha_{\max}$ by a chain of successive neighbours. This definition being equivalent to the fact that there exists a neighbour of the current pixel which has already been accepted.

In the following figure, the pixels are stored by connectivity in abscissa and by gradient magnitude in ordinate. Then a pixel of the edge e_1 would be accepted since its magnitude is above $\alpha_{\max}$. A pixel of e_4 is rejected since its magnitude is below $\alpha_{\min}$. A pixel in e_3 would be rejected since all the pixels connected to it have a magnitude below $\alpha_{\max}$. Finally, a pixel of e_2 with magnitude between $\alpha_{\max}$ and $\alpha_{\min}$ would be accepted since there exists pixels connected to it which have a magnitude above $\alpha_{\max}$.

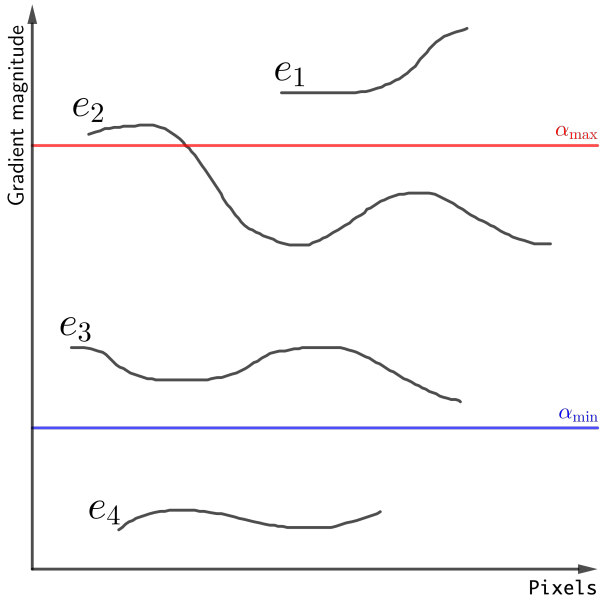

Figure 10: Hysteresis thresholding.

Of course this thresholding method forces us to choose two new parameters $\alpha_{\max}$ and $\alpha_{\min}$. Although there are ways to set these parameters automatically. For example, we can deduce them from the median magnitude of the gradient. Let us denote m the median magnitude and $m_{\max}$, $m_{\min}$ respectively the maximum magnitude and minimum magnitude. Then we can compute the thresholds by setting,

$$\alpha_{\min} = \min(m_{\min}, (1-s) * m), \text{ and } \alpha_{\max} = \max(m_{\max}, (1+s) * m).$$

We still have to choose a value for the parameter s. Based on people expertise the value $s = \frac{1}{3}$ seems to be good in most of cases. Obviously this is not a proof and the choice for the value of s would have to be considered carefully.

In the following figure we show different results of the Canny filter depending on the chosen thresholds. The wide thresholds correspond to the parameters

$$\alpha_{\min} = m_{\min} + (m_{\max} - m_{\min}) \times 0.05 \text{ and } \alpha_{\max} = m_{\max} - (m_{\max} - m_{\min}) \times 0.05.$$

The narrow thresholds correspond to

$$\alpha_{\min} = m_{\min} + (m_{\max} - m_{\min}) \times 0.45 \text{ and } \alpha_{\max} = m_{\max} - (m_{\max} - m_{\min}) \times 0.45.$$

Automatic thresholds correspond to the previous strategy.

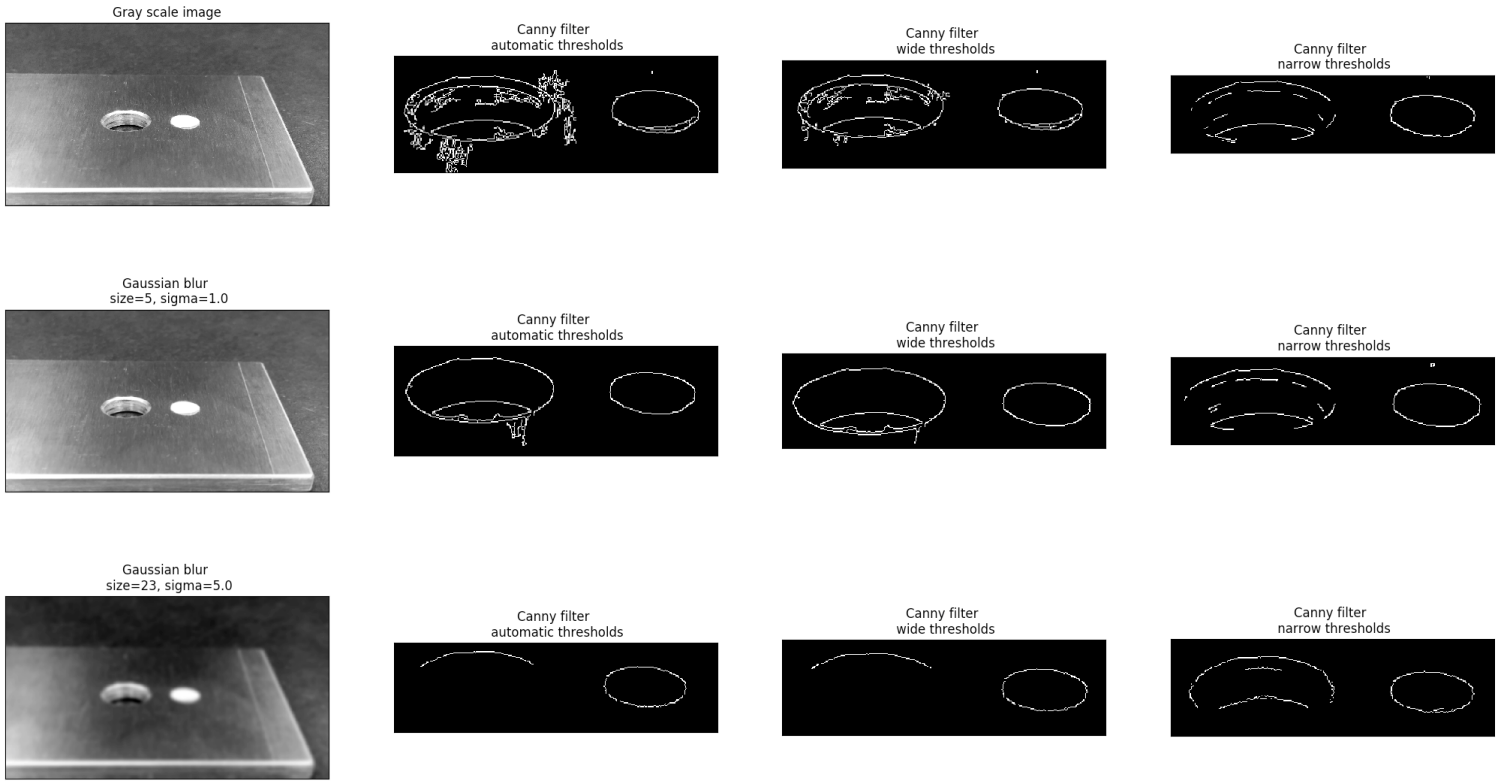

Figure 11: Examples of Canny filters.

1.2.1 Summary on Canny filter

In figure 11 we clearly see that the Canny filter is very sensitive to noise filtering. Also in this particular example the filter seems to be less sensitive to the values of thresholds. Of course these conclusions are very specific to this particular example. To handle the general case it would be necessary to develop a method to estimate optimal choices for the following parameters: k and σ^2 the size and variance of the Gaussian filter, the size of the Sobel derivatives kernels, $\alpha_{\min}$ and $\alpha_{\max}$ or s the hysteresis thresholding parameters.

Also we have presented here the classical version of the Canny filter but there are still recent publications which improve the method. For example in [6] they replace the Gaussian noise filter by statistical methods and the hysteresis thresholding by a genetic algorithm, leading to a new version of Canny filter which seems to give better results.

Moreover, as we can see in figure 11 we have not captured all the information we wanted about the hole. In fact, in the original picture we can see that the entry of the hole has a chamfer due to the drilling. Consequently, the diameter we want to measure is not the diameter of the ellipse at the top of the chamfer but right below the chamfer.

In the following figure, we see in red the edges we wanted our algorithm to detect. On the bottom it is the result of our Canny filter (in Gaussian blur of size 5 and variance 1 case).

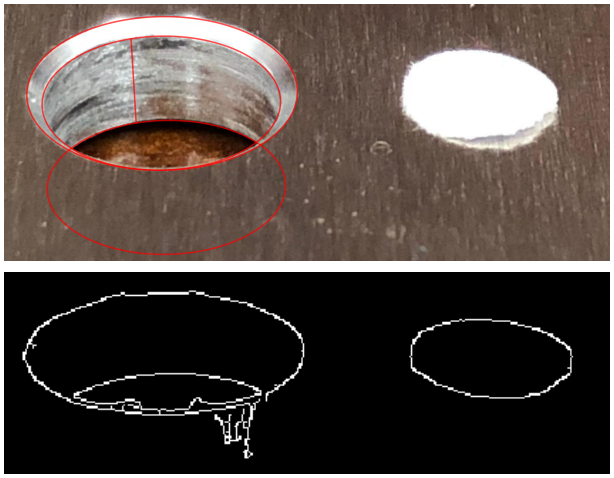

Figure 12: Comparison between the edges of interest and the Canny filter result.

As we can see on this figure our algorithm only sees the top ellipse at the entry of the hole. Consequently if we measure the diameter using this ellipse we will get an error about double the width of the chamfer which is around 0.5 millimetres. This problem shows how important are the choices of the parameters in the Canny filter. In our case we have to be able to distinguish the cases where there is a chamfer or not and choose the filter parameters in order to be able to distinguish the ellipse at the top of the chamfer and the one at the bottom.

Here is an example of the result we could expect from Canny filter with correctly chosen parameters.

Figure 13: Example of correct result.

Once we get this result we still have to find a way to partition this result to distinguish all the different edges and to be able to measure the diameter of the ellipse and the depth of the hole (in pixels).

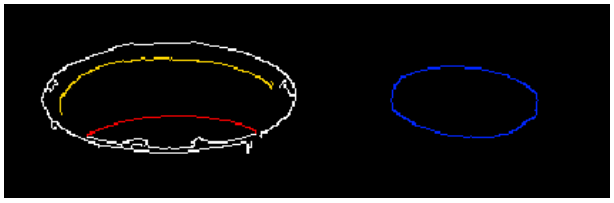

Figure 14: Example of partition which identifies the interesting parts of the picture.

To achieve this goal we could imagine an algorithm walking along the different edges, detecting if they are connected (or not) by a branching point and distinguish them according to this branching point. In the following we will assume that our algorithm can give us a result as in figure 14.

1.3 Dimensions measurements

As we mentioned in 1.1 we are just considering the case of female parts with circular holes. We developed two different methods to compute the diameter and depth of the hole, and since the resolution of the picture is important for the accuracy of the computations, we tested both methods two times with the same picture. The first test in both cases was made with a low resolution version of the original picture, and the second test with the original resolution of the picture given by a smartphone. For the implementation of the methods, we used a paper circle of 55 millimetres diameter (approximately) to emulate a round sticker as a size reference in the picture, and with the help of the open-source raster graphics editor GIMP, we used the laplacian edge detection to process the images. Hence, what we see in the processed images is on the right side an "ellipse" corresponding to the size reference, and on the left side two "ellipses" corresponding to the top of the hole and between these "ellipses" a ring that corresponds to the chamfer, and at the bottom of the left side of the picture we can appreciate a piece of an "ellipse" corresponding to the bottom of the hole. For both methods, we will compute the diameter of the hole using the ellipse on the bottom of the chamfer. Notice that we do not obtain real ellipses in the formal mathematical sense, since they are perturbed after the image processing, which justifies the use of the notation "ellipses" in the last paragraph.

1.4 Description of the first method

By drawing a horizontal line passing through the centre of the perturbed ellipses corresponding to the size reference and the bottom of the chamfer, we compute the

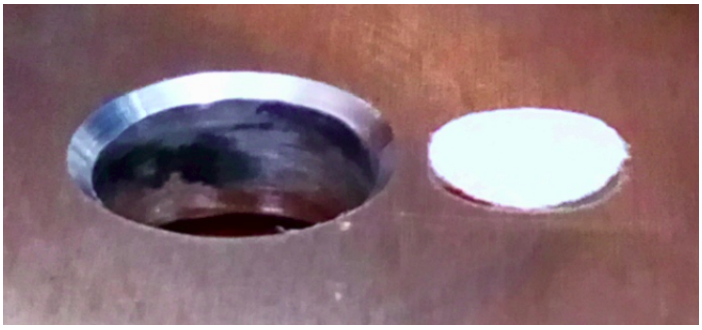

Figure 15: Normal resolution picture

Figure 16: Low resolution picture

length of the diameters of the hole and the size reference in pixels and by using a cross-multiplication we obtain the real diameter of the hole since the real length of the size reference is known. In other words, we have the formula

$$\text{hole diameter in cm.} = \frac{\text{size reference diameter in cm.} \times \text{hole diameter in pixels}}{\text{size reference diameter in pixels}}.$$

Finally, by drawing a vertical line and taking the segment of this line corresponding to the depth of the hole we can compute the depth in pixels and using the same process as before we obtain the depth of the hole (depth between the chamfer) with the following formula

$$\text{hole depth in cm.} = \frac{\text{size reference diameter in cm.} \times \text{hole depth in pixels}}{\text{size reference diameter in pixels}}.$$

1.5 Description of the second method

In this method we change the perturbed ellipses by real ellipses, in the formal mathematical sense in such a way that the real ellipses are close enough to the ellipses we see on the original picture obtained by the perspective of the camera. With that purpose we will use the next result.

Theorem 1. *Five points in general position in the plane determine a unique conic passing through these points.*

With the help of the geometry software GeoGebra, we select five points on each one of the perturbed ellipses corresponding to the reference point and the bottom of the chamfer. Therefore, by Theorem 1 these points determine two ellipses. An easy manipulation with GeoGebra allows us to compute the axis of both ellipses and as in the first method we obtain the real diameter of the hole by using a cross-multiplication given by the formula

$$\text{hole diameter in cm.} = \frac{\text{size reference diameter in cm.} \times \text{mayor-axis of the hole-ellipse}}{\text{mayor-axis of the size reference-ellipse}}.$$

Finally, once again with the help of GeoGebra we select five points on the piece of the perturbed ellipse corresponding to the bottom of the hole. Notice that even if on the picture we can only see a little piece of a perturbed ellipse, we can define a complete ellipse by Theorem 1. Hence, by drawing the line that passes through the centre of the ellipses determined by the top and the bottom of the hole and considering the segment of this line corresponding to the depth of the hole, we obtain the next formula as before by using a cross-multiplication

$$\text{hole depth in cm.} = \frac{\text{size reference diameter in cm.} \times \text{hole depth measured in GeoGebra}}{\text{mayor-axis of the size reference-ellipse}}.$$

1.6 Correcting the depth

Notice that the depth computed in both methods is not quite accurate. In Figure 17 we have a two dimensional representation of the situation. We have a blue line representing the direction in which the picture was taken, a red line representing the picture line, i.e. what we see on the picture, and the segments DB and BE representing the depth and the size reference respectively. Hence, the depth computed in both methods is not the real depth but the length of the segment AB.

Now, remark that on the one hand by trigonometric identities we have

$$\text{depth} = \frac{AB}{\sin \angle ADB} = \frac{\text{measured depth}}{\sin \angle ADB},$$

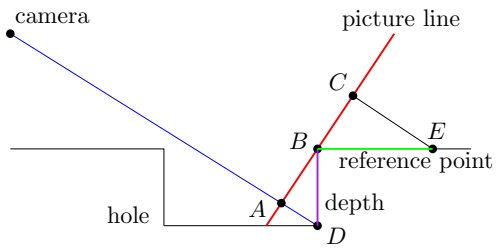

Figure 17: Two dimensional representation

and on the other hand, since the segments BC and BE represent the minor and major axis respectively of the ellipse representing the size reference, we get by trigonometric identities that

$$\cos \angle CBE = \frac{BC}{BE} = \frac{\text{minor-axis of the size reference ellipse}}{\text{major-axis of the size reference ellipse}}.$$

Finally, since the angles $\angle ADB$ and $\angle CBE$ are equal, we conclude that

$$\text{depth} = \frac{\text{measured depth}}{\sqrt{1 - \left(\frac{\text{minor-axis of the size reference ellipse}}{\text{major-axis of the reference point ellipse}}\right)^2}}.$$

1.7 Results

We present the results obtained, and as we can see the best results obtained are with the second method with the normal resolution picture, which is good news since the second method is easier to describe in computational terms.

Picture low resolution, first method	Reference point	Hole
Real diameter (cm)	0.55	0.83
Real depth (cm)		0.4
Measured diameter in pixels	24	30
Measured depth in pixels		12
Computed diameter (cm)		0.6875
Diameter absolute error (cm)		0.1425
Diameter relative error (%)		17.1686747
Measured vertical diameter in pixels	12	
Cos (angle CBE)	0.5	
Computed depth (cm)		0.275
Depth corrected (cm)		0.317542648
Depth absolue error (cm)		0.082457352
Depth relative error (%)		20.61433799

Figure 18: Low resolution picture, first method

Picture normal resolution, first method	Reference point	Hole
Real diameter (cm)	0.55	0.83
Real depth (cm)		0.4
Measured diameter in pixels	200	276
Measured depth in pixels		107
Computed diameter (cm)		0.759
Diameter absolute error (cm)		0.071
Diameter relative error (%)		8.554216867
Measured vertical diameter in pixels	100	
Cos (angle CBE)	0.5	
Computed depth (cm)		0.29425
Depth corrected (cm)		0.339770633
Depth absolue error (cm)		0.060229367
Depth relative error (%)		15.05734165

Figure 19: Normal resolution picture, first method

Picture low resolution, second method	Reference point	Hole
Real diameter (cm)	0.55	0.83
Real depth (cm)		0.4
Measured major-axis (ggb unit)	0.38	0.47
Measured depth (ggb unit)		0.2
Computed major-axis (cm)		0.680263158
Major-axis absolue error (cm)		0.149736842
Major-axis relative error (%)		18.04058339
Measured minor-axis (ggb unit)	0.21	
Cos (angle CBE)	0.5526315789	
Computed depth (cm)		0.289473684
Depth corrected (cm)		0.347329938
Depth absolue error (cm)		0.052670062
Depth relative error (%)		13.16751553

Figure 20: Low resolution picture, second method

Picture normal resolution, second method	Reference point	Hole
Real diameter (cm)	0.55	0.83
Real depth (cm)		0.4
Measured major-axis (ggb unit)	0.27	0.37
Measured depth (ggb unit)		0.15
Computed major-axis (cm)		0.753703704
Major-axis absolue error (cm)		0.076296296
Major-axis relative error (%)		9.192324855
Measured minor-axis (ggb unit)	0.13	
Cos (angle CBE)	0.4814814815	
Computed depth (cm)		0.305555556
Depth corrected (cm)		0.34862613
Depth absolue error (cm)		0.05137387
Depth relative error (%)		12.84346748

Figure 21: Normal resolution picture, second method

Future improvements

We have presented our first approach to tackle this particular industrial problem. We purposely skipped both proprietary solutions and complex methods to develop our

own, simple approach, tailored for what we thought was ARaymond's need. The core idea was to seek for a balance between computational simplicity and accuracy, trying to propose a satisfactory solution with a simple method. It might not be accurate enough as is, depending on the industrial standard at stake, and it certainly lacks the "male part" matching part, but it paves the way for more research in this direction. The numerical results presented here are not precise enough to tell if our method is competitive or not, although with a proper implementation they could become relevant and, if needed, help to improve the method. We can already draw a short list of potential improvements. First, a proper implementation of the method is needed. Using Python and OpenCV, such an implementation would not be difficult to create. Then, the method (especially in the edge detection part) is based on several unknown parameters. Consequently these parameters need to be estimated. This would definitely increase the computational complexity of the program but is a crucial step to improve accuracy. Finally, this method need to be tested, namely to study the approximation error.

References

[1] G Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[2] John Canny. A Computational Approach to Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698, nov 1986.

[3] G. Deng and L.W. Cahill. An adaptive Gaussian filter for noise reduction and edge detection. *1993 IEEE Conference Record Nuclear Science Symposium and Medical Imaging Conference*, pages 1615–1619.

[4] R O Duda and P E Hart. *Pattern Classification and Scene Analysis*. Wiley, 1973.

[5] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, New York, NY, USA, 2 edition, 2003.

[6] Ruiyuan Liu and Jian Mao. Research on Improved Canny Edge Detection Algorithm. 03053:3–6, 2018.

[7] Glauco V. Pedrosa and Celia A Z Barcelos. Anisotropic diffusion for effective shape corner point detection. *Pattern Recognition Letters*, 31(12):1658–1664, 2010.