Improving Edge Detection In Highly Noised Sheet-Metal Images

Javier Gallego-Sánchez, Jorge Calera-Rubio
Departamento de Lenguajes y Sistemas Informáticos
Universidad de Alicante, Apt.99, E-03080 Alicante, Spain
{jgallego, calera}@dlsi.ua.es

Abstract

This article proposes a new method for robust and accurate detection of the orientation and the location of an object on low-contrast surfaces in an industrial context. To be more efficient and effective, our method employs only artificial vision. Therefore, productivity is increased since it avoids the use of additional mechanical devices to ensure the accuracy of the system.

The technical core is the treatment of straight line contours that occur in close neighbourhood to each other and with similar orientations. It is a particular problem in stacks of objects but can also occur in other applications. New techniques are introduced to ensure the robustness of the system and to tackle the problem of noise, such as an auto-threshold segmentation process, a new type of histogram and a robust regression method used to compute the result with a higher precision.

1. Introduction

Our aim is to accurately detect the location and the orientation of objects so as to transmit this data to industrial robots. The objects that we are working with are rectangular, but the algorithm allows other types of straight edged shapes. These objects are stacked up and always are placed in roughly the same location. The main problem is that although their location is known, as they are stacked up, they could be gently moved or rotated. It causes a lot of noise just around the areas of interest, due to faint edges: overlapping edges, little projections or parallel laterals (better explained in section 2).

Production lines use an intermediate system, called *squaring machine*, to obtain the precise location of these objects. This process is very slow, because the robotic arm, after taking a sheet from the stack, has to deposite it in this machine to obtain the correct orientation, and then to pick it up again. To be more efficient and effective, our method only employs artificial vision to solve this problem. Therefore, it accelerates the whole process (as shown in the experimentation), since it avoids the necessity of this additional mechanical device.

Accuracy and robustness are fundamental in the area of industrial applications, because it directly influence the results of a subsequent processing. There are also several problems and requirements inherent in these types of environments, such as the multiplicity of experimental situations, the lighting, the shadows or the possible defects of the objects. Moreover, the system has associated another major difficulty: The lens of the camera produce a radial distortion which change the appearance of the image, so that straight lines appear to be curve. In addition, it is a generic problem in which the objects can have different sizes, shapes and colors.

For all these reasons, we can not soley rely on the results of basic techniques. The proposed algorithm is designed specifically for this type of problem, but it can have many other applications as proposed in the conclusions. The following section describes the proposed method in details. Section 3 shows the results of the experimentation and a comparison with other methods. The last section presents the conclusions and the lines for future work.

2. Description of the Algorithm

The main problem in detecting the position and the orientation of stacked objects is that they are usually slightly moved. It may seem that the topmost sheet occludes all others. But in general, a minimal variation creates a bundle of such lines just around the areas of interest. In addition, there may be dirt or burrs caused by the production of the sheets close to the margin or highlights from the cutting or punching deformations. From now on, all these lines will be referred as lines of noise (see Figure 1).

The following algorithm is proposed to detect the location and the orientation of the topmost object of a stack of objects:

1. *Capture* an image and *correct* the distortion. All the images are taken from the top using controlled lighting in order to improve imaging conditions.

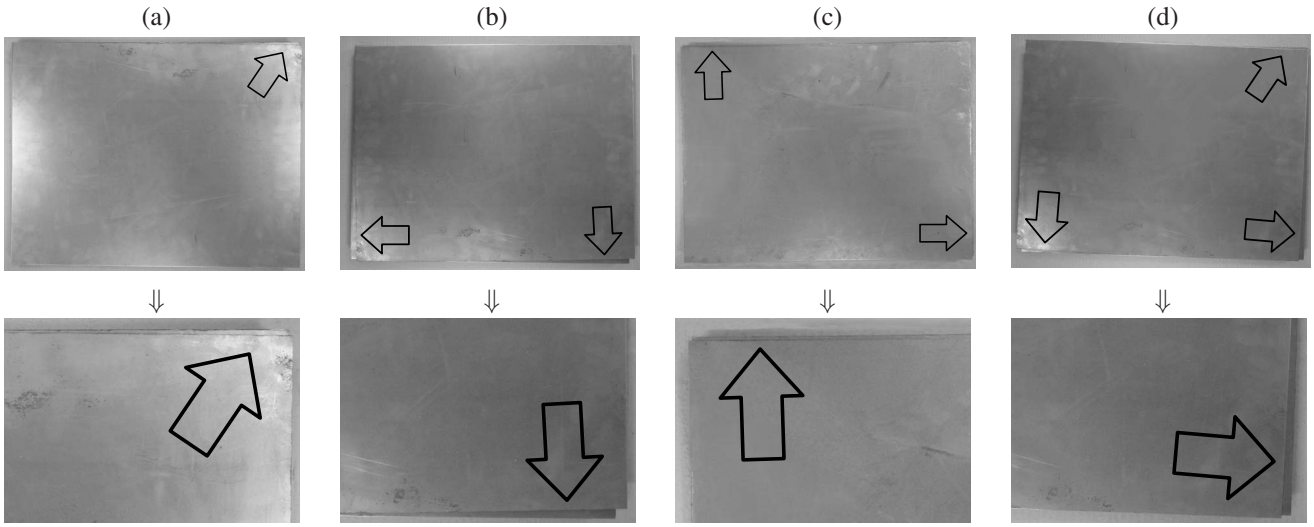

Figure 1. First row: Different images of the used datatest (see section 3). Second row: Detail of the above image in which appears more lines of noise (indicated by an arrow).

2. *Iterative edge-based segmentation with auto-threshold*.

3. *Hough transform*: It is applied to find the straight lines of the object. The extracted lines are also *filtered* and *classified*.

4. *Weighted union of nearly-coincident lines*: It joins the discontinued lines.

5. *Accumulated size histogram*: It adjusts the lines that belong to each lateral.

6. *Calculate the representative line of each lateral*: To perform this task a robust regression estimator is used.

Steps 1 and 3 are known techniques, but properly tuned to match the specific problem requirements. In steps 2, 4, 5 and 6 new algorithms are introduced. Each step is described in details in next sections.

2.1. Calibration and Undistortion

Lots of research has been done in this area (in [10] can be seen a review). We have used the calibration algorithm proposed by Zhang [10]. It uses a two-dimensional calibration pattern, and it does not need to know the movement of the camera. Moreover, the modifications proposed in [7] have been applied. They improve the results using different pre and post-processes, and separating in two parts the calculation of the parameters. If all the parameters are calculated at the same time, the results would be worse. It is due to the fact that the error in a variable could be offset by other errors. The steps that this algorithm follows are:

1. Take n images of the calibration pattern from different positions.

2. Detect the reference marks of the calibration pattern with subpixel precision.

3. Preprocess the distortions and normalize the coordinates of the points [1, 3].

4. Estimate the matrix of the intrinsic camera parameters [10].

5. Compute the extrinsic parameters given the intrinsic parameters [10].

6. Calculate the undistortied image using the homografy matrix H.

This algorithm reduces the average error of the estimation to 0.053 ± 0.012 (in points of the image, using the same resolution as in [10]). It improves the results of the basic algorithm of Zhang, which obtains an error of 0.345 ± 0.059. This error is calculated using the average of geometric distances between the set of measured points and the marks of the pattern projected with the estimated parameters [7].

To reduce the error of calibration, the following considerations are proposed: Take about 10 pictures in which the pattern covers the whole image. According to [9], there is no improvement in the results with more than 8 images. The same camera parameters have to be maintained for the calibration and for all the images that are going to be undistortied.

2.2. Iterative Segmentation with Auto-Threshold

A Canny segmentation algorithm is applied iteratively over the undistortied image until the gray level reaches the

value of δ. This value is an empirically calculated percentage of the level of border pixels that the segmented image should have (In our case, it is set to $\delta = 2.5\%$ and $\xi = 0.5\%$).

```
th := lastThreshold;
do {
   img := Canny(img, th);
   δc := CalculateGrayLevel(img);
   if( δc > δ ) th++;
   else         th--;
} while( | δc - δ | > ξ );
lastThreshold := th;
```

Other edge-based segmentation algorithms have been tried, but due to variability (lighting, shadows, defects, colours) better results are obtained by applying a simple method iteratively until the desired threshold is reached. In this way, it obtains more robustness to possible changes in brightness and lighting.

2.3. Hough Transform

Hough transform [8] is applied to extract the straight lines that appear after the segmentation. We use the classical algorithm which is based on a probabilistic approach to find straight lines. We have also tried some variations of the classical algorithm, such as multi-scale or fuzzy Hough, obtaining similar results. Therefore, it is preferred to use a simple and fast method to extract the segments. The threshold values are set reaching a compromise, because, if it removes a lot of noise, it may also remove real edges (In our case, they are set to: $threshold = 30$, $minimum - length = 50$, and $maximum - gap = 20$).

The extracted segments are filtered and classified using the following criteria:

1. *Orientation*: A range of possible orientations $[\theta_1, \theta_2]$ is established ($\theta_R \pm 10$, where θ_R is the reference orientation of the object). The lines with a different orientation are filtered. Overlapping lines are also filtered, because Hough algorithm sometimes obtains more than one result for the same line.

2. *Location*: It is used to:

   (a) *Classify*: Each line is classified in the set $\mathcal{M}_k$ for its correspondig lateral k (For this type of objects: $0 \leq k < 4$).

   (b) *Filter*: All the lines whose location is outside of the limits are removed. These limits are obtained from the known dimensions of the object (see section 3) and the range of allowed error in the displacement ($\mathcal{R}_k \pm 0.05$m, where $\mathcal{R}_k$ is the expected position for the lateral $\mathcal{M}_k$).

2.4. Weighted Union of Nearly-Coincident Lines

This process is applied to join discontinued lines, and form a new line l_j with a new orientation θ_j. The new angle is calculated by weighting the angle of each line by its size, thereby, it gives more weight to the orientation of long lines and forms more solid lines. θ_j is obtained using $\theta_j = \sum \frac{\theta_i \cdot d(l_i)}{d_t} \; \forall l_i \in \mathcal{N}$, where $d(l_i) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$, $d_t = \sum d(l_i) \; \forall l_i \in \mathcal{N}$ and $\mathcal{N}$ is the set of lines which fulfills all the following criteria:

- Lines with the same classification (Section 2.3).
- Coincident lines with the same slope and y-interception. However, a small angle error $|\theta_1 - \theta_2| \leq \phi$ is permited (In our case, it is set to $\phi = 0.05$). This way, it also allows the union of lines that are nearly coincident.
- Lines within a certain range of distance ($d_m \leq max(d(l_i))/30 \; \forall l_i \in \mathcal{M}_k$). It uses the *point-line distance* equation $d_m = \frac{\vec{AB} \times \vec{AC}}{|\vec{AB}|}$, from the line AB to the midpoint C of the line to join.

All the lines which keep having a size smaller than a given threshold $\mathcal{U}$ are filtered (It is set to $\mathcal{U} = max(d(l_i))/20 \; \forall l_i \in \mathcal{M}_k$). These lines are mainly created by points of noise, so it often has a wrong orientation.

It would be possible to adjust the Hough parameters to be less restrictive, but this post-processing obtains better results because: First, with a more restrictive parameters, it ensures that the extracted lines actually appear in the image. Second, this process takes into account the possible angle errors, allowing a small error in the union, and also adjusting the angle of the resulting line.

2.5. Accumulated Size Histogram

An histogram is calculated for each of the main orientations of the object (vertical and horizontal). It establishes a relation between the lines' location and their size. To calculate the vertical histogram (Fig. 2(a)) the equation 1 is used, where d_k is the accumulated size of all the lines in the set L_k, $L_k = \{l_i : k \leq b_i < k + 1, \; 0 \leq k < img_{Width}\}$ (Fig. 2(c)) and $l_i \equiv x = a_i y + b_i$. Thus, the histogram is divided into ranges of $[k, k+1]$ which accumulate the size of all the lines that appear in them. For the horizontal histogram (Fig. 2(b)) the same equation is used, but $l_i \equiv y = a_i x + b_i$ and $0 \leq k < img_{Height}$.

$$d_k = \sum d(l_i) \;\; \forall \, l_i \in L_k \tag{1}$$

We have also tried other types of histogram, such us accumulating the size at the midpoint of each segment. But the current filter obtains better results because it takes into

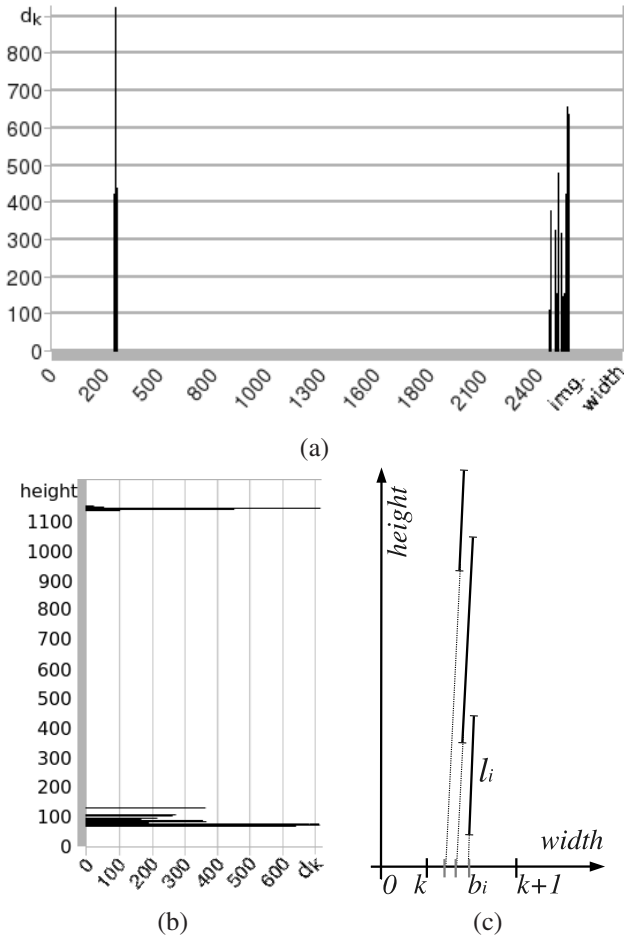

Figure 2. (a, b) Accumulated-size histogram for the vertical and the horizontal lines. (c) Size-accumulation scheme for the range $[k, k+1]$.

account the main orientation of each lateral and the orientation and the position of each individual line.

These histograms are used to remove the outliers (lines of noise) in the distribution. First, the standard deviation σ is calculated for each mode. Then, the interquartile range (IQR) is defined as $IQR = 1,35\sigma$ [5] and its limits are set as lower quartile $Q_1 = \bar{x} - IQR/2$ and upper quartile $Q_3 = \bar{x} + IQR/2$. All the lines which are outside the range $[f_1, f_2] = [Q_1 - 1,5IQR,\ Q_3 + 1,5IQR]$ are considered outliers, and therefore are removed. In other words, the lateral's frequency distribution is compared with an ideal distribution (a perfect edge represented by a single line) which identifies the atypical values in the distribution.

2.6. Calculate the Representative Line of each Lateral

To calculate the representative line of each side, instead of using a couple of points of each line, a sampling of the filtered lines at a constant distance τ is used. Thus, more weight is applied to long lines. τ is calculated as $\tau = gcd(d(l_i)) \ \ \forall\ l_i \in \mathcal{M}_k$. The sampling equation is derived from the formula of the Euclidean distance $\tau^2 = (x_2 - x_1)^2 + (y_2 - y_1)^2$ and the straight line's equation $y_2 = ax_2 + b$. The combination of these two formulas returns two solutions that correspond to the previous and the next point.

This sampling process returns a two-dimensional sample (x_1, y_1), (x_2, y_2),..., (x_n, y_n) for each lateral. The intention is to approximate this distribution by the line which minimizes the sum of quadratic differences between the observed values and the adjusted by the line [6]. The problem is that the *linear model* provides an estimator which is very sensitive to the possible presence of abnormal data and homoscedasticity. *Robust regression* is a powerful alternative which obtains better results with a slightly higher computational cost. It is insensitive to small departures from the idealized assumptions for which a statistical estimator is optimized [4]. The basic idea is to calculate the estimator α which minimizes the function:

$$\Psi(\alpha) = \sum_{i=1}^{n} \omega_\alpha(e_i) e^2 \tag{2}$$

where ω is a weighted function that is introduced to reduce (or even remove) the effect of the outliers. Thus, the weights $\omega(e_i)$ are defined to take small values for the high remainders e_i. There are many estimators that can be used for this purpose. In this case, as the error is not normally distributed, it is better to use a maximum-likelihood formula for the parameter estimation [6].

The independent variable is changed depending on the orientation of the lateral. The vertical and the horizontal laterals will be adjusted to the line which minimizes the error $e_i^2 = (x_i - \hat{x}_i)^2$ and $e_i^2 = (y_i - \hat{y}_i)^2$ respectively.

Therefore, it obtains good results without the need of using a *multiple regression model*, much more complex and time-comsuming. It has also been compared with other robust methods, such as iterative re-weighting [6] and RANSAC [3] (see section 3), obtaining similar results but with a higher computational cost. Moreover, the solutions obtained with these alternative methods may not be the optimal one and the convergence is not guaranteed.

2.7. Calculating the Final Location and Orientation

The last step is to calculate the correlation coefficient R of each representative line. It is a measure of the linear relationship among variables. An absolute value of $|R|$ close to 1 indicates a strong linear relationship. Therefore, it allows to identify the most appropriate lateral to calculate the final orientation of the object (primary lateral). To calculate the final location and orientation, it is only necessary to know

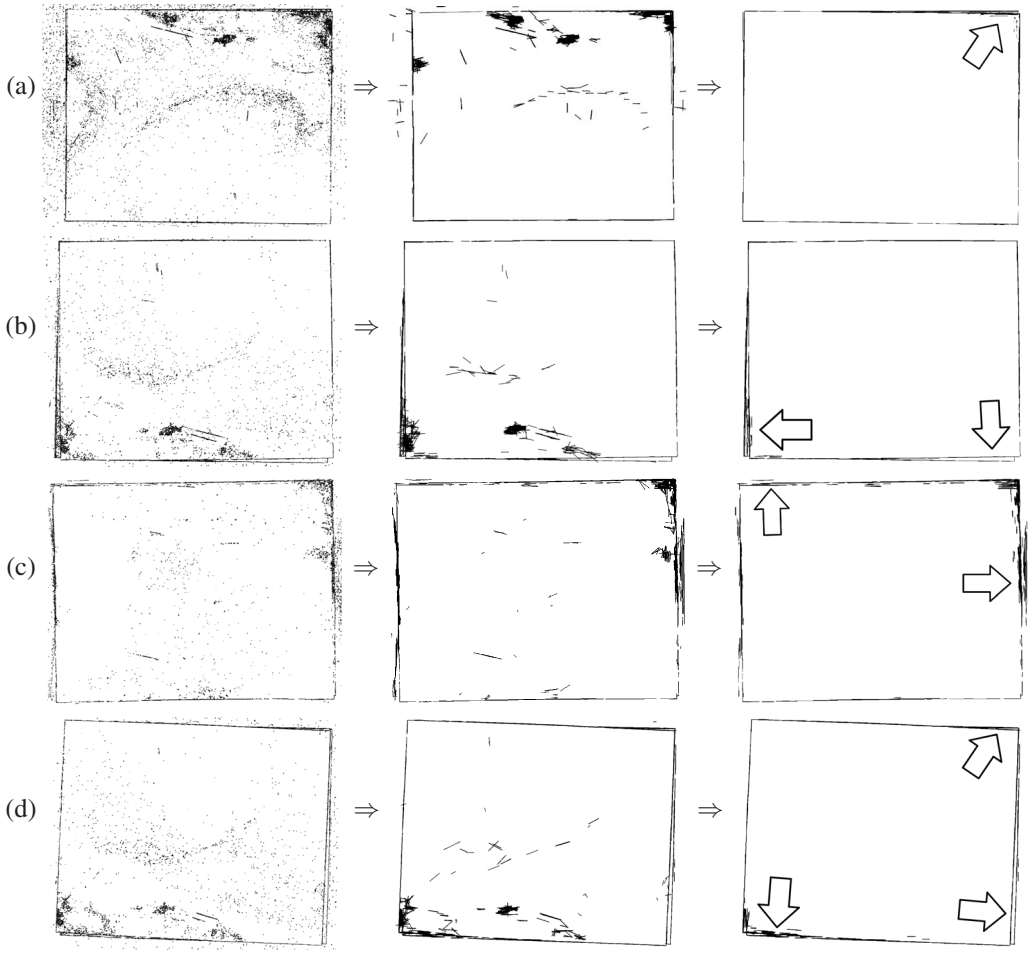

Figure 3. Steps of the process for the objects in Figure 1: Iterative segmentation, Hough transform and filtering. The parts with more lines of noise are indicated by arrows.

with precision one lateral and one corner. The location of the primary corner is calculated from the intersection between the primary lateral and the perpendicular lateral with the highest correlation coefficient. In the example of Figure 3, the chosen laterals are: (a) bottom-right, (b) right-top, (c) bottom-left and (d) top-left.

The location of the rest of laterals can be estimated geometrically. First, the equivalence in pixels of the object's size has to be calculated. The conversion factor $\lambda =\{$ *CCD width in meters / Image width in pixels* $\}$ is used to convert pixels into metres. To calculate the CCD width, its dimensions $(1/3'')$ and its proportions $(\frac{x}{y} = \frac{4}{3})$ are needed. Therefore, the value of x (total width in metres) can be isolated from the equation $x^2 + y^2 = (\frac{1}{3}'')^2$. Next, the object's size is calculated using the general equation of lens $\frac{1}{s} + \frac{1}{s'} = \frac{1}{f}$, where s is the object distance, s' is the image distance and f is the focal length [2].

3. Results

In order to test the algorithm, a database of 200 images with a resolution of 3072×2304 pixels has been used[1]. These images are obtained from a top view of a stack of 50 sheets (similar to Fig. 1). In each image, slight variations in lighting and in the position and the orientation of the sheets are introduced. The dimensions of the sheet metal are $50 \times 40 \times 0.1$ cm. The obtained results have been compared with very precise measures taken manually over the image.

Table 1 shows the results of the proposed method. It is also compared with previous results obtained using: RANSAC, iterative re-weighting (explained in section 2.6), Linear Regression, Robust Regression (RR) without sampling, RR without filtering and a simple Model-Based Technique (First, the edges closest to the center are detected, then they are used in this order to find the object that matches the model). In addition, other methods have been tried getting worse results, this is due to the high variability

[1]If you want to obtain this database, please contact the first author.

of the problem (see explanation in the Introduction section).

Table 1 shows the average error of the primary lateral orientation and of the primary corner location. It also shows the percentage of hits obtained during the selection of the most appropriate lateral (see section 2.7). The proposed method obtains better results with a similar computational cost. Its average angle error is 0.144°, with a minimum error of 0.02°and a maximum error of 0.25°.

The complete algorithm has an average computational cost of 0.37 sec. It is linear with respect to the size of the image. The squaring machine takes ∼15 sec. in performing the same task (as described in the introduction). So, if the total manipulation time by the robot is ∼60 sec., by incorporating the proposed system, productivity would be increased in ∼20 items per hour.

Method	Average angle error	Average corner error
Proposed method	0.144°	0.00142 m
RANSAC	0.162°	0.00146 m
Iterative re-weighting	0.171°	0.00149 m
Linear regression	0.181°	0.00153 m
RR without sampling	0.986°	0.0087 m
Model-based technique	2.43°	0.0194 m
RR without filtering	3.67°	0.0256 m

Method	Correct lateral %
Proposed method	95.7%
RANSAC	94.2%
Iterative re-weighting	93.9%
Linear regression	93.6%
RR without sampling	89.1%
Model-based technique	-
RR without filtering	78.6%

Table 1. Benchmarking

4. Conclusions and Future Work

A new method for accurate and robust detection of the location and the orientation of an object on low-contrast surfaces has been presented. It has been designed specifically for this type of problem, in which each step is focused to ensure the reliability and the accuracy of the system. It is a particular problem in stacks of objects but can also occur in other applications, such as industrial automation, control of measures, quality control, positioning of objects, and packaging. Moreover, the algorithm allows other types of straight edged shapes, sizes and colors.

According to the results, the proposed method could be incorporated into an operating environment to replace the squaring machine, thereby substantially increasing the productivity of the entire production line.

Acknowledgments

The authors thank the reviewers for their thoughtful comments which helped to improve the initial manuscript. This work is partially supported by Spanish MICINN (contract TIN2009-14205-C04-01, contract DPI2006-15542-C04-01) and CONSOLIDER-INGENIO 2010 (contract CSD2007-00018).

References

[1] M. Ahmed and A. Farag. Nonmetric calibration of camera lens distortion: differential methods and robust estimation. *Image Processing, IEEE Transactions on*, 14(8):1215–1230, Aug. 2005.

[2] O. Faugeras. *Three-dimensional computer vision: a geometric viewpoint*. The MIT Press. Cambridge, Massachusetts, 1993.

[3] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2004.

[4] P. J. Huber. *Robust Statistics*. Wiley Series in Probability and Statistics, 1981.

[5] I. Michael Sullivan. *Fundamentals of Statistics*. Pearson Prentice Hall, 2006.

[6] W. H. Press, B. P. Flanner, S. A. Teukolsky, and W. T. Vetterling. *Numerical Recipes: The Art of Scientific Computing*. Cambridge University Press, 2007.

[7] C. Ricolfe-Viala and A.-J. Sanchez-Salmeron. Improved camera calibration method based on a two-dimensional template. In *Pattern Recognition and Image Analysis*, pages 420–427, 2007.

[8] L. Shapiro and G. Stockman. *Computer Vision*. Prentice-Hall, Inc, 2001.

[9] W. Sun and J. R. Cooperstock. An empirical evaluation of factors influencing camera calibration accuracy using three publicly available techniques. *Machine Vision and Applications*, 17:51–67, 2006.

[10] Z. Zhang. A flexible new technique for camera calibration. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000.