ELLIPSE DETECTION IN DIGITAL IMAGE DATA USING GEOMETRIC FEATURES

Lars Libuda
Chair of Technical Computer Science
Aachen University, Germany
libuda@techinfo.rwth-aachen.de

Ingo Grothues
Chair of Technical Computer Science
Aachen University, Germany
grothues@techinfo.rwth-aachen.de

Karl-Friedrich Kraiss
Chair of Technical Computer Science
Aachen University, Germany
kraiss@techinfo.rwth-aachen.de

Keywords: Ellipse Detection, Shape Analysis & Representation, Image Analysis.

Abstract: Ellipse detection is an important task in vision based systems because many real world objects can be described by this primitive. This paper presents a fast data driven four stage filtering process which uses geometric features in each stage to synthesize ellipses from binary image data with the help of lines, arcs, and extended arcs. It can cope with partially occluded and overlapping ellipses, works fast and accurate and keeps memory consumption to a minimum.

1 INTRODUCTION

The detection of ellipses in digital image data is an important task in vision based systems as shapes of real world objects can often be described by geometric primitives like ellipses or be assembled by them (Sanz et al., 1988; Radford and Houghton, 1989).Applications include but are not limited to gaze tracking (Canzler and Kraiss, 2004), ball tracking in soccer games (d'Orazio et al., 2004), vehicle detection (Radford and Houghton, 1989), cell counting in breast cancer cell samples (Mclaughlin, 1998) or traffic sign detection (Piccioli et al., 1994).

Algorithms for ellipse detection have to cope with noisy image data and partially occluded ellipses and they also have to produce accurate results as fast as possible to be suitable for realtime applications. Furthermore, memory usage should be low, since ellipse detection is mostly just a preprocessing step for algorithms applied in later stages.

Ellipses are described by 5 parameters: center point (x_E, y_E), two semi-axes (a, b), and orientation α. The best known method to estimate these parameters is the standard Hough transform (Duda and Hart, 1972) and its derivatives, e. g. (Xu et al., 1990). Special versions of Hough transforms adapted to ellipse extraction also exists (Ho and Chen, 1996; Guil and Zapata, 1997). There is however a common disadvantage: Hough transforms demand a trade off between processing speed and accuracy and consume a lot of memory. This led to the development of methods independent of any Hough transform. McLaughlin (McLaughlin and Alder, 1998) proposed an algorithm called "UpWrite" for ellipse detection. It works faster and more accurate than the above mentioned methods, it fails however in case of partially occluded ellipses. The latter problem is addressed by Kim et al. (Kim et al., 2002). They introduced a two-stage reconstruction algorithm which is able to detect partially occluded ellipses but do not treat memory consumption.

The algorithm presented in this paper can be added to the category of Hough transform independent algorithms. Ellipse detection is regarded as a data driven four stage filtering process (Fig. 1). The first stage extracts short straight lines from a binary input im-

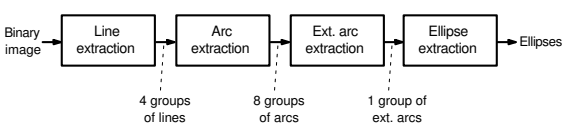
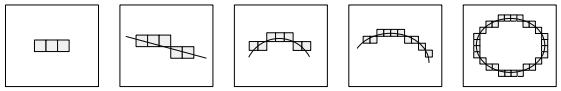


Figure 1: Ellipse detection as four stage filtering process.



(a) Segment (b) Line (c) Arc (d) Ext. arc (e) Ellipse

Figure 2: Extracted objects during ellipse detection

age which is created with Canny's algorithm (Canny, 1986). In the second stage, these lines are combined to small arcs which are synthesized to extended arcs in the third stage. Extended arcs are finally used to create ellipses. Each stage uses geometric features of the extracted objects to synthesize them from objects extracted in the previous stage.

The remaining part of this paper is structured as follows: Section 2 gives an overview on the entire filtering process and basic definitions. Section 3 describes the process according to Fig.1 in detail. First results and performance of the algorithm are presented in section 4.

2 OVERVIEW AND DEFINITIONS

The elements extracted in each processing stage are depicted in Fig. 2. A segment consists of at least two adjacent pixels and belongs to one of the line orientation groups denoted in Fig. 3. Within each separate group of segments lines are synthesized from adjacent segments which do not exceed a predefined quantization error with regard to the ideal analogue line represented by these segments. An arc is created from at least two adjacent lines of one line orientation group. The lines must not exceed a given error in the tangents to an estimated circle which these lines represent. During arc extraction each line orientation group is split in two arc orientation groups (Fig. 4) depending on the arc's orientation with respect to the ellipse's midpoint. Extended arcs consist of three adjacent arcs from consecutive arc orientation groups. Finally an ellipse is constructed from one ore more extended arcs which describe the same ellipse with a predefined tolerance and cover the circumference of the described ellipse to a predefined degree.

During the filtering process it is necessary to access the base objects of a constructed element. Therefore, each synthesized object keeps a reference to all base

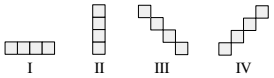


Figure 3: Line groups

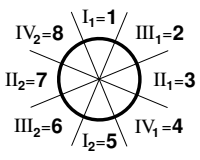


Figure 4: Arc groups

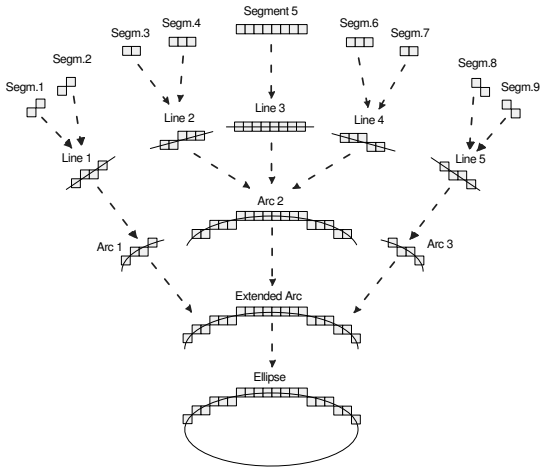


Figure 5: Extracted elements and their relationships

objects it is composed of. Fig. 5 visualizes this concept which makes it possible to trace back each extracted object down to the single pixels belonging to this object.

3 ELLIPSE DETECTION

This section describes the used algorithms and the geometric features of the extracted objects in the single filter stages. All algorithms are described for orientation group I only. The same algorithms may be applied to all other groups after rotating the pixel coordinates by $\pm 45°$ and $90°$ respectively.

3.1 Line extraction

In the first step lines are extracted from the binary input image using the algorithm proposed by Kim (Kim et al., 2003). The algorithm outputs for each line orientation group $g \epsilon [I, II, III, IV]$ a set of n_g lines $\mathbf{LS}_g = \{\underline{\mathbf{L}}_i, i = 1..n_g\}$ with $\underline{\mathbf{L}}_i = (x_{si}, y_{si}, x_{ei}, y_{ei}, x_{Mi}, y_{Mi}, \Theta_i)$ describing the start position (x_{si}, y_{si}), end position (x_{ei}, y_{ei}), midpoint (x_{Mi}, y_{Mi}), and slope Θ_i of each line. The last three elements are calculated by the following equations:

$$x_{Mi} = \frac{x_{si}+x_{ei}}{2} \quad , \quad y_{Mi} = \frac{y_{si}+y_{ei}}{2}$$

$$\Theta_i = \tan^{-1}\left(\frac{y_{si}-y_{ei}}{x_{ei}-x_{si}}\right)$$

3.2 Arc extraction

The second processing stage combines lines to small arcs for each line set $\mathbf{LS}_g$. The algorithm selects a target line $\underline{\mathbf{L}}_i$ from $\mathbf{LS}_g$ and stores it in an empty arc

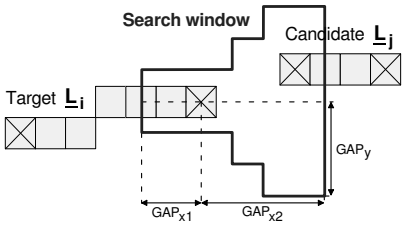


Figure 6a: Search window for arcs

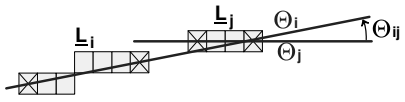


Figure 6b: Intersection angle

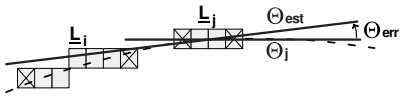


Figure 6c: Tangent error

line set $\mathbf{LA}$. Subsequently it searches for a candidate line $\underline{\mathbf{L}}_j$ within an adaptive triangular search window (Fig. 6a). With the predefined maximum distance D_{line} the window parameters are calculated by:

$$\begin{aligned} GAP_{x1} &= \frac{x_{ei} - x_{si} + 1}{2} - 1 \\ GAP_{x2} &= D_{line} \\ GAP_y &= D_{line} - 1 \end{aligned}$$

If a candidate line $\underline{\mathbf{L}}_j$ is found $\underline{\mathbf{L}}_i$ and $\underline{\mathbf{L}}_j$ will be considered parts of the same arc if they satisfy the following two conditions:

1. The intersection angle $\Theta_{ij} = |\Theta_i - \Theta_j|$ has to be in the range $0° \leq \Theta_{ij} \leq 45°$ (Fig. 6b).
2. The error of Θ_j compared to the estimated circle tangent Θ_{est} in the midpoint $(x_{Mj},\, y_{Mj})$ of $\underline{\mathbf{L}}_j$ must not exceed a given angle tolerance $\Theta_{err,line}$ (Fig. 6c). By using all lines in $\mathbf{LA}$ and the candidate line $\underline{\mathbf{L}}_j$ the circle midpoint $(\tilde{x}_C,\, \tilde{y}_C)$ and its radius $\tilde{R}$ are estimated with the help of Thomas' algorithm (Thomas and Chan, 1989). Now, Θ_{est} can be calculated and the condition checked by:

$$\Theta_{est} = \tan^{-1}\left(\frac{x_{Mj} - \tilde{x}_C}{\tilde{y}_C - y_{Mj}}\right)$$

$$|\Theta_j - \Theta_{est}| \;<\; \Theta_{err,line}$$

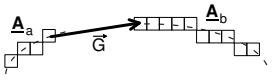
If $\underline{\mathbf{L}}_j$ satisfies all conditions it is added to $\mathbf{LA}$ and a new iteration starts with $\underline{\mathbf{L}}_j$ as the new target. If $\underline{\mathbf{L}}_j$ is not found or fails either test and $\mathbf{LA}$ contains more than one element, a new arc is found. In this case the final circle parameters x_C, y_C and R are estimated from the lines contained in $\mathbf{LA}$ and stored in a vector $\underline{\mathbf{A}} = (\mathbf{LA}, x_C, y_C, R)$. Depending on the arc's position to the estimated circle midpoint, it is assigned to one of two possible arc groups (see Fig. 4). Afterwards the algorithm chooses a new target line from $\mathbf{LS}_g$ which is not already part of an arc and starts at the beginning. It terminates when all lines have been visited.

After application of this algorithm to all line sets the result is a set of n_g arcs $\mathbf{AS}_g = \{\underline{\mathbf{A}}_i, i = 1..n_g\}$ with $\underline{\mathbf{A}}_i = (\mathbf{LA}_i, x_{Ci}, y_{Ci}, R_i)$ for each group $g \,\epsilon\, [1..8]$.

3.3 Extended arc extraction

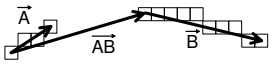
In the third step arcs are combined to extended arcs. This is necessary because arcs are too small for an accurate ellipse estimation. For one extended arc three adjacent arcs $\underline{\mathbf{A}}_a$, $\underline{\mathbf{A}}_b$ and $\underline{\mathbf{A}}_c$ of consecutive arc groups have to be found. This can be achieved by selecting a target arc $\underline{\mathbf{A}}_b$ of the arc set $\mathbf{AS}_g$ and searching the sets $\mathbf{AS}_{g-1}$ and $\mathbf{AS}_{g+1}$ for the candidate arcs $\underline{\mathbf{A}}_a$ and $\underline{\mathbf{A}}_c$. To ensure that target and candidates describe the same ellipse, several conditions are checked. Conditions 1-3 apply to both arc pairs a/b and b/c, but are described for a/b only. Conditions 4-6 apply to all three arcs a/b/c. For arc pairs we define the gap vector $\overrightarrow{G}$ pointing from the endpoint of one arc to the start point of the other. We define $\measuredangle(\underline{\mathbf{a}}, \underline{\mathbf{b}})$ to be the angle between the vectors $\underline{\mathbf{a}}$ and $\underline{\mathbf{b}}$.

1. The **absolute distance** of the arcs in horizontal and vertical direction given by $|G_x|$ and $|G_y|$ must not exceed the predefined maximum distance D_{arc}.



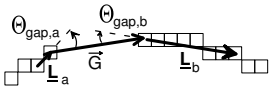
$$|G_x| \leq D_{arc}$$
$$|G_y| \leq D_{arc}$$

2. The **relative distance** d_{rel} of the arcs must be greater than the predefined minimum d_{min}. Vector $\overrightarrow{AB}$ connects the arc startpoints and $\overrightarrow{A}$, $\overrightarrow{B}$ are vectors pointing from start- to endpoint of each arc.



$$d_{rel} = \frac{|\overrightarrow{AB}|}{|\overrightarrow{A}|} \;>\; d_{min}$$

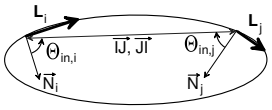
3. The **gap angles** of both arcs must be less than the predefined maximum $\Theta_{gap,max}$. With $\underline{\mathbf{L}}_a$ as last line of arc $\underline{\mathbf{A}}_a$ and $\underline{\mathbf{L}}_b$ as first line of $\underline{\mathbf{A}}_b$ the gap angles can be calculated as the angles between these lines and $\overrightarrow{G}$.



$$\Theta_{gap,a} = \measuredangle(\underline{\mathbf{L}}_a, \overrightarrow{G})$$
$$\Theta_{gap,b} = \measuredangle(\underline{\mathbf{L}}_b, \overrightarrow{G})$$

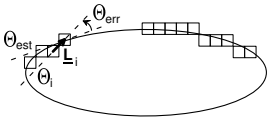
4. The **inner angles** of all arc line pairs must be less than 90°. Let $\mathbf{LX}$ be the set of all lines of $\underline{\mathbf{A}}_a, \underline{\mathbf{A}}_b, \underline{\mathbf{A}}_c$ and $\underline{\mathbf{L}}_i, \underline{\mathbf{L}}_j$ be two lines of $\mathbf{LX}$. The inner angles are the angles between their normal vectors $\overrightarrow{N_i}$, $\overrightarrow{N_j}$ and

their startpoint connection $\overrightarrow{IJ}$ and $\overrightarrow{JI}$ respectively.



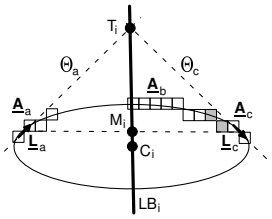
$$\Theta_{in,i} = \measuredangle(\overrightarrow{N_i}, \overrightarrow{IJ})$$
$$\Theta_{in,j} = \measuredangle(\overrightarrow{N_j}, \overrightarrow{JI})$$

5. The **tangent error** of all arc lines compared to the estimated ellipse must be less than the predefined maximum $\Theta_{err,arc}$. Using $\underline{\mathbf{A}}_a$, $\underline{\mathbf{A}}_b$, $\underline{\mathbf{A}}_c$ the ellipse parameters $\tilde{x}_E$,$\tilde{y}_E$,$\tilde{a}$,$\tilde{b}$,$\tilde{\alpha}$ are estimated with the algorithm proposed in (Fitzgibbon and Fisher, 1999). For each arc line $\underline{\mathbf{L}}_i$ we compare the line tangent Θ_i and the ellipse tangent Θ_{est} in the midpoint (x_{Mi}, y_{Mi}).



$$\Theta_{est} = \tan^{-1}\left(\frac{-\tilde{b}^2 \cdot x_{Mi}}{\tilde{a}^2 \cdot y_{Mi}}\right)$$
$$|\Theta_i - \Theta_{est}| \quad < \quad \Theta_{err,arc}$$

6. The **line beam** of the three extended arcs must run within the maximum distance D_{LB} from their estimated ellipse center point $C_i = (\tilde{x}_{Ei}, \tilde{y}_{Ei})$. With $\underline{\mathbf{L}}_a$ being the first line of $\underline{\mathbf{A}}_a$ and $\underline{\mathbf{L}}_c$ being the last line of $\underline{\mathbf{A}}_c$ the line beam LB_i can be calculated as the line passing through the points $\mathbf{T}_i$ and $\mathbf{M}_i$, whereas $\mathbf{T}_i$ is the intersection point of the line tangents Θ_a and Θ_c and $\mathbf{M}_i$ is the midpoint of the connection of the midpoints of $\underline{\mathbf{L}}_a$ and $\underline{\mathbf{L}}_c$.



If all three arcs satisfy all conditions, a new extended arc $\underline{\mathbf{X}}$ is created. After all arcs have been visited the result of this stage is a set of n extended arcs $\mathbf{XS} = \{\underline{\mathbf{X}}_i, i = 1..n\}$ with $\underline{\mathbf{X}}_i = (\underline{\mathbf{A}}_{ai}, \underline{\mathbf{A}}_{bi}, \underline{\mathbf{A}}_{ci}, \tilde{x}_{Ei}, \tilde{y}_{Ei}, \tilde{a}_i, \tilde{b}_i, \tilde{\alpha}_i, LB_i)$.

3.4 Ellipse extraction

In the last step extended arcs are used to create ellipses. The algorithm merges extended arcs $\underline{\mathbf{X}}_i$ belonging to the same ellipse $\underline{\mathbf{E}}_j$ to a set $\mathbf{XE}_j = \{\underline{\mathbf{X}}_i, i = 1..n\}$ in three steps. Merged extended arcs $\underline{\mathbf{X}}_i$ are removed from the set $\mathbf{XS}$ because they can be part of one ellipse only.

Because each extended arc consists of three arcs, adjacent extended arcs can overlap in up to two arcs. In the first step, these overlapping arcs are identified by searching extended arcs composed of identical arcs. The identified objects are then checked by three conditions whether they describe the same ellipse:

1. The tangent error of all arc lines compared to the jointly estimated ellipse must be less than $\Theta_{err,arc}$. This is identical to condition 5 in section 3.3.
2. The line beams of all extended arcs have to intersect within the maximum distance D_{LB} from the ellipse center point.
3. The ellipse contour mismatch of the start- and endpoints (x_i, y_i) of all arc lines must not exceed the predefined maximum $\delta_{ell,max}$ and is checked by:

$$\left| \left(\frac{x_i}{\tilde{a}}\right)^2 + \left(\frac{y_i}{\tilde{b}}\right)^2 - 1 \right| \quad < \quad \delta_{ell,max}$$

In the second step non-overlapping extended arcs are taken from $\mathbf{XS}$ and it is tried to assign them to one of the merge sets $\mathbf{XE}_j$ created in the first step. An extended arc has to fulfill the same conditions 1-3 to become part of a set $\mathbf{XE}_j$.

The third step tries to merge the remaining extended arcs in $\mathbf{XS}$. The algorithm compares the ellipse parameters of each extended arc and merges those that match with a predefined accuracy. The ellipse center must not differ more than D_{match} and the semi-axis have to match with a relative percentage r_{match}.

Finally the ellipse parameters x_{Ej}, y_{Ej}, a_j, b_j, and α_j are calculated for every merge set $\mathbf{XE}_j$. Subsequently, the ellipse circumference C_j is approximated by:

$$C_j \approx \pi\left(1.5\,(a_j + b_j) - \sqrt{a_j b_j}\right)$$

If the set of all arc lines in $\mathbf{XE}_j$ covers C_j to a predefined percentage C_{min}, the ellipse is added to the final ellipse set $\mathbf{ES} = \{\underline{\mathbf{E}}_j, j = 1..n\}$ with $\underline{\mathbf{E}}_j = (\mathbf{XE}_j, x_{Ej}, y_{Ej}, a_j, b_j, \alpha_j)$.

4 RESULTS

This section presents results which were obtained on a Pentium 4 system with 2.8 GHz. Fig. 6 visualizes all intermediate results during ellipse extraction on an outdoor real world image of size 800 x 600 pixels. Although there is lots of clutter in the binary edge image all relevant ellipses are found and no false positives are detected. However, false positives may be detected in case lines form a partial ellipse which actually do not belong to a real world ellipse as demonstrated in the top row of Fig. 7. This happens because lines and the derived structures are the only information used to search for ellipses. By incorporating more knowledge in the detection process, e.g. color or texture analysis within an ellipse candidate, false positives can be reduced. This is one task for future work.

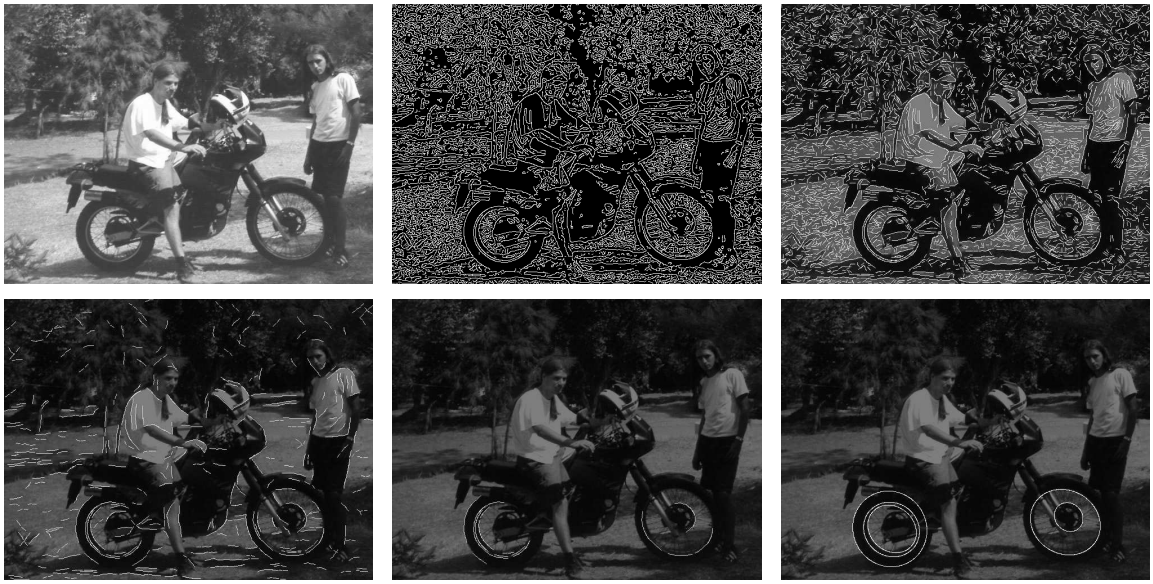


Figure 6: Visualization of the different steps during ellipse detection in an outdoor real world image. *From top left to bottom right*: Input image, binary edge image, lines, arcs, extended arcs and ellipses. The extracted elements are superimposed on the input image (800 x 600 pixels).

Table 1: Memory consumption and processing time for image (800 x 600 pixels) in Fig. 6.

Object	Quantity	Memory usage	Time
Segments	89295	2092.9 KByte	62 ms
Lines	9094	931.8 KByte	172 ms
Arcs	540	63.7 KByte	47 ms
Ext. arcs	21	3.6 KByte	12 ms
Ellipses	5	0.3 KByte	3 ms
Total		3092.3 KByte	296 ms

Tab. 1 summarizes the amount of extracted objects in each filter stage, the memory consumption and processing time for the image in Fig.6. Each processing stage reduces the number of processed objects approximately by one order of magnitude, which means that the time consuming checks are only applied to very few objects. Memory is allocated only for the extracted objects. Of course memory consumption and processing time depend on the complexity of the original image. On an image of size 320 x 240 pixels the average processing time is 45 ms. However, with a more sophisticated preprocessing which narrows the search space, speed can be increased and memory consumption can be decreased even further. This optimization is the second task for future work.

The two bottom rows of Fig.7 show example images demonstrating the detection of overlapping and partially occluded ellipses. All ellipses are found in all images but their accuracy depends on the amount of their visible circumference. The more data is available for one ellipse the more precise are its estimated parameters.

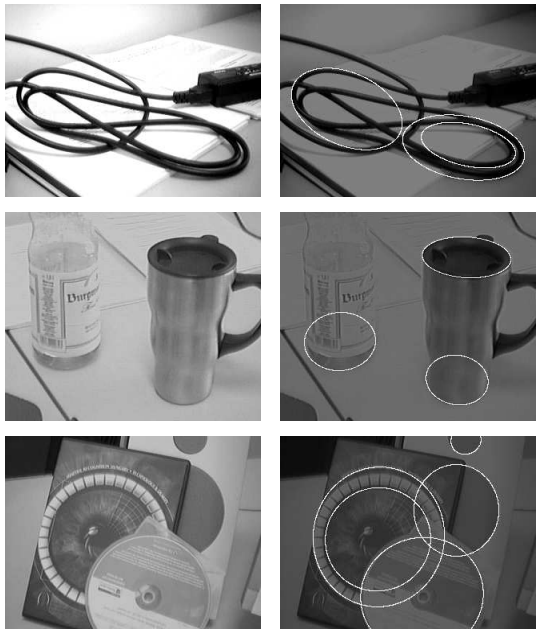


Figure 7: Examples for the detection of false positives (top row) and overlapping and partially occluded (bottom rows) ellipses in real world images. All results were obtained with the same parameter settings (see Tab. 2).

Finally, the number of parameters introduced in section 3 has to be discussed. On the one hand many

Table 2: Importance of parameters. The 3^{rd} column shows the values used to obtain the results in Fig. 7.

Symbol	Importance	Value	Ref.
D_{line}	-	3 pix.	3.2
$\Theta_{err,line}$	+	18.0°	3.2
D_{arc}	+	37 pix.	3.3
d_{min}	-	0.47	3.3
$\Theta_{gap,max}$	-	30.0°	3.3
$\Theta_{err,arc}$	+	14.0°	3.3, 3.4
D_{LB}	o	4 pix.	3.3, 3.4
$\delta_{ell,max}$	-	2.7	3.4
D_{match}	o	5.0	3.4
r_{match}	-	0.8	3.4
C_{min}	+	0.25	3.4

parameters allow to adapt the algorithm to nearly all situations but on the other hand it is sometimes hard to find the optimal configuration. For the latter case we ranked the parameters to identify the important once. Tab. 2 shows which parameters should be changed first to adapt the algorithm in case it does not produce the desired results with its default settings. Parameters marked with "+" are most important for an adaption and have to be changed first. Parameters marked with "o" can be used for fine tuning the results and parameters marked with "-" do not influence the final results. They can be replaced by constant values in future versions of the algorithm. In this way only four parameters remain which is a fair amount for an algorithm of this complexity. However, all ellipses in this paper were found using the same parameter settings.

5 CONCLUSION

This paper introduces a fast and robust algorithm for ellipse extraction from binary image data based on a four stage data driven filtering process. The obtained results support the conclusion that it is able to cope with partially occluded ellipses and noisy image data. It produces accurate results and keeps memory consumption to a minimum. Future work includes the incorporation of more knowledge, e.g. color information, to distinct between real ellipses and false positives and speed optimization. The algorithm is available as open source in the LTI-LIB project at http://ltilib.sourceforge.net.

REFERENCES

Canny, J. (1986). A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698.

Canzler, U. and Kraiss, K.-F. (2004). Person-adaptive facial feature analysis for an advanced wheelchair user-interface. In Drews, P., editor, *Conference on Mechatronics & Robotics*, volume Part III, pages 871–876, Aachen. Sascha Eysoldt Verlag.

d'Orazio, T., Guaragnella, C., Leo, M., and Distante, A. (2004). A new algorithm for ball recognition using circle hough transform and neural classifier. *Pattern Recognition*, 37(3):393–408.

Duda, R. and Hart, P. (1972). Use of the hough transformation to detect lines and curves in pictures. *Communications of the ACM*, 15(1):11–15.

Fitzgibbon, A. W.and Pilu, M. and Fisher, R. B. (1999). Direct least-squares fitting of ellipses. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(5):476–480.

Guil, N. and Zapata, E. (1997). Lower order circle and ellipse hough transform. *Pattern Recognition*, 30(10):1729–1744.

Ho, C. and Chen, L. (1996). A high-speed algorithm for elliptical object detection. *IEEE Transactions on Image Processing*, 5(3):547–550.

Kim, E., Haseyama, M., and Kitajima, H. (2002). Fast and robust ellipse extraction from complicated images. In *Proceedings of the first International Conference on Information Technology & Applications*, Bathurst, Australia.

Kim, E., Haseyama, M., and Kitajima, H. (2003). Fast line extraction from digital images using line segments. *Systems and Computers in Japan*, 34(10):76–89.

Mclaughlin, R. (1998). Randomized hough transform: Improved ellipse detection with comparison. *Pattern Recognition Letters*, 19(3-4):299–305.

McLaughlin, R. and Alder, M. (1998). The Hough transform versus the UpWrite. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(4):396–400.

Piccioli, G., Michelli, E., Parodi, P., and Campani, M. (1994). Robust road sign detection and recognition from image sequences. In *Proceedings of the IEEE Symposium on Intelligent Vehicles*, pages 278–283, Paris, FR.

Radford, C. and Houghton, D. (1989). Vehicle detection in open-world scenes using a hough transform technique. In *Third International Conference on Image Processing and its Applications*, pages 78–82, Warwick, UK.

Sanz, J., Hinkle, E., and Jain, A. (1988). *Radon and Projection Transform-Based Computer Vision*. Springer Verlag.

Thomas, S. and Chan, Y. (1989). A simple approach for the estimation of circular arc center and its radius. *Computer Vision, Graphics, and Image Processing*, 45(3):362–370.

Xu, L., Oja, E., and Kultanen, P. (1990). A new curve detection method: Randomized hough transform (rht). *Pattern Recognition Letters*, 11(5):331–338.