PClines – Line Detection Using Parallel Coordinates

Markéta Dubská, Adam Herout, and Jiří Havel
Graph@FIT
Brno University of Technology
Božetěchova 2, Brno
idubska,herout,ihavel@fit.vutbr.cz

Abstract

Detection of lines in raster images is often performed using Hough transform. This paper presents a new parameterization of lines and a modification of the Hough transform – PClines. PClines are based on parallel coordinates, a coordinate system used mostly or solely for high-dimensional data visualization. The PClines algorithm is described in the paper; its accuracy is evaluated numerically and compared to the commonly used line detectors based on the Hough transform. The results show that PClines outperform the existing approaches in terms of accuracy. Besides, PClines are computationally extremely efficient, require no floating-point operations, and can be easily accelerated by different hardware architectures.

1. Introduction

The Hough transform is a well-known tool for detecting shapes and objects in raster images. Originally, Hough [10] defined the transformation for detecting lines; later it was extended for more complex shapes, such as circles, ellipses, etc., and even generalized for arbitrary patterns [1].

When used for detecting lines in 2D raster images, the Hough transform is defined by a *parameterization* of lines: each line is described by two parameters. The input image is preprocessed and for each pixel which is likely to belong to a line, voting accumulators corresponding to lines which could be coincident with the pixel are increased. Next, the accumulators in the parameter space are searched for local maxima above a given threshold, which correspond to likely lines in the original image. The Hough transform was formalized by Princen et al. [15] and described as an *hypothesis testing* process.

Hough [10] parameterized the lines by their *slope* and y-axis *intercept*. A very popular parameterization introduced by Duda and Hart [5] is denoted as θ-ϱ; it is important for its inherently bounded parameter space. It is based on a line equation in the normal form: $y \sin\theta + x \cos\theta = \varrho$. Parameter θ represents the angle of inclination and ϱ is the length of the shortest chord between the line and the origin of the image coordinate system. Wallace [16] introduced a different bounded parameterization, the so-called *Muff-transform*. As the basis for his parameterization, a bounding rectangle around the image is used. Each line intersects the bounding box at exactly two points and their distances from the origin along the perimeter are used as the parameters. Using a circle instead of a rectangle defines another bounded parameterization, called *fan-beam parameterization* [14]. Again, a line and a circle intersect at exactly two points. Angles defined by these two points were used for line parameterization for the first time by Eckhardt and Maderlechner [6]. The Muff transform is also a basis for parameterization introduced by Forman [8], who combined it with θ-ϱ and represented lines by the first intersect point and the line's orientation θ.

A subset of all possible line parameterizations are *point-to-line mappings* (PTLM), where a point in the source image corresponds to a line in the Hough space and – naturally for the Hough transform – a point in the Hough space represents a line in the x-y image space. Point-to-line mappings were studied by Bhattacharya et al. [2], who proved that for a single PTLM, the Hough space must be infinite. However, for any PTLM, a complementary PTLM can be found so that the two mappings define two finite Hough spaces containing all lines possible in the x-y image space.

Parallel coordinates (PC) were invented in 1885 by Maurice d'Ocagne [4] and further studied and popularized by Alfred Inselberg [12]. Currently, parallel coordinates are mostly used as a tool for visualization of multidimensional data. This paper presents "PClines", a new parameterization of lines based on parallel coordinates which can be used for line detection by the Hough transform. An analytical and experimental evaluation of the parameterization is given, demonstrating that the new parameterization outperforms the currently used ones in several aspects.

Basic information on the parallel coordinates is reviewed in Section 2. The new line parameterization PClines is pre-

sented and formalized in Section 3. Section 4 contains experimental evaluation of line detection based on the PClines parameterization, comparing it to the currently used solutions. Conclusions and ideas for further development of PClines are given in Section 5.

2. Parallel Coordinates

A common way to visualize vectors in N-dimensional Euclidean space is by using Cartesian coordinates, where each vector in a given N-dimensional vector space is represented by exactly one point in the N-dimensional coordinate system. However, visualization of spaces with more than two dimensions on two-dimensional drawing equipment (paper, computer monitor, etc.) is only possible by projection, which – especially for higher dimensionalities – might not be intuitive and can severly restrict the amount of information presented at a time.

The parallel coordinate system represents the vector space by axes which are mutually parallel. Each N-dimensional vector is then represented by $N-1$ lines connecting the axes – see Fig. 1. In this text, we will be using a Euclidean plane with a u-v Cartesian coordinate system to define positions of points in the space of parallel coordinates. For defining these points, a notation $(u, v, w)_{\mathbb{P}^2}$ will be used for homogeneous coordinates in the projective space $\mathbb{P}^2$ and $(u, v)_{\mathbb{E}^2}$ will be used for Cartesian coordinates in the Euclidean space $\mathbb{E}^2$.

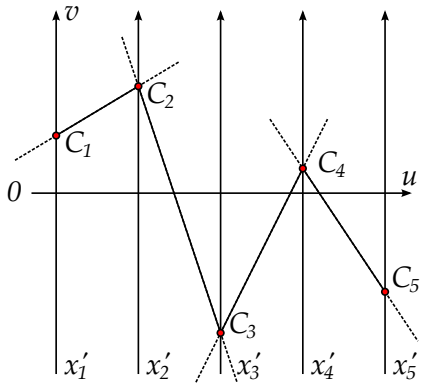

Figure 1. Representation of a 5-dimensional vector in parallel coordinates. The vector is represented by its coordinates $C_1, \ldots, C_5$ on axes $x'_1, \ldots, x'_5$, connected by a complete polyline (composed of 4 infinite lines).

In the two-dimensional case, points in the x-y space are represented as lines in the space of parallel coordinates. Representations of collinear points intersect at one point – the representation of a line (see Fig. 2). Based on this relation, it is possible to define a point-to-line mapping between these spaces. For some cases, such as line $\ell : y = x$, the corresponding point $\overline{\ell}$ lies in infinity (it is an ideal point). Projective space $\mathbb{P}^2$ (contrary to the Euclidean $\mathbb{E}^2$ space)

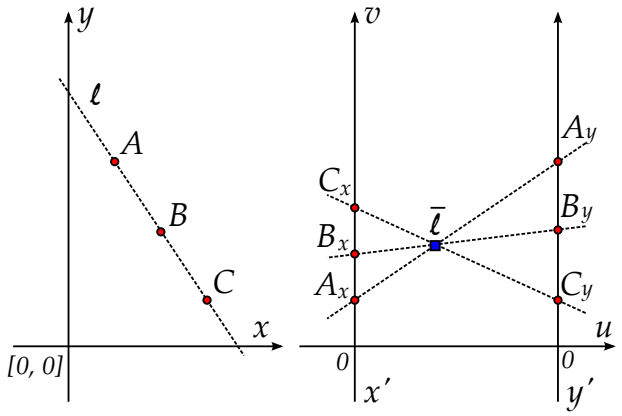

Figure 2. Three collinear points in parallel coordinates: (left) Cartesian space, (right) space of parallel coordinates. Line ℓ is represented by point $\overline{\ell}$ in parallel coordinates.

provides coordinates for these special cases. A relation between line $\ell : ax + by + c = 0$ (denoted as $[a, b, c]$) and its representing point $\overline{\ell}$ can be defined by mapping:

$$\ell : \left[a, b, c\right] \rightarrow \overline{\ell} : \left(db, -c, a + b\right)_{\mathbb{P}^2}, \qquad (1)$$

where d is the distance between parallel axes x' and y'.

This can be generalized for higher dimensions since a line in $\mathbb{E}^N$ is a set of all points satisfying system of $(N-1)$ linearly independent linear equations. The equations can be represented by $(N-1)$ points in the space of parallel coordinates – see Fig. 3. Different spaces of parallel coor-

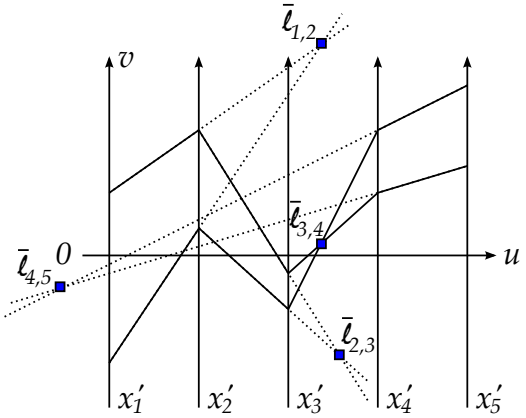

Figure 3. Points $\overline{\ell}_{i,i+1}$ exactly define a set of 4 equations which determine a line in a 5-dimensional space.

dinates can be defined by using the axes in a different order. All these parameterizations are equivalent, but study of their relations is outside of the scope of this paper (please refer to [12] instead).

In a 2D space, line ℓ coincides with point A in the x-y space if and only if its image $\overline{\ell}$ lies on line $\overline{A}$ in the space of parallel coordinates. This means that line $\overline{A}$ defines a common intersection of all such lines ℓ – this property is discussed in Sec. 3.2. In spaces with higher dimensionalities

(> 2), where the lines can also be skewed, not just parallel or intersecting each other, parallel coordinates of two lines can be used for calculating the distance between them.

3. Line Detection Using Parallel Coordinates

The PC representation of line $\ell : y = mx + b$ in the u-v space is $\overline{\ell} = (d, b, 1-m)_{\mathbb{P}^2}$, where d is the distance between the parallel axes x' and y'. The line's representation $\overline{\ell}$ is between the axes x' and y' if and only if $-\infty < m < 0$. For $m = 1$, $\overline{\ell}$ is an ideal point (a point in infinity). For $m = 0$, $\overline{\ell}$ lies on the y' axis, for vertical lines ($m = \pm\infty$), $\overline{\ell}$ lies on the x' axis.

Besides this space of parallel coordinates x', y' (further referred to as *straight*, $\mathcal{S}$), we propose using a *twisted* ($\mathcal{T}$) system $x', -y'$, which is identical to the straight space, except that the y axis is inverted. In the twisted space, $\overline{\ell}$ is between the axes x' and $-y'$ if and only if $0 < m < \infty$. By combining the *straight* and the *twisted* spaces, the whole $\mathcal{TS}$ plane can be constructed, as shown in Fig. 4. Figure 4

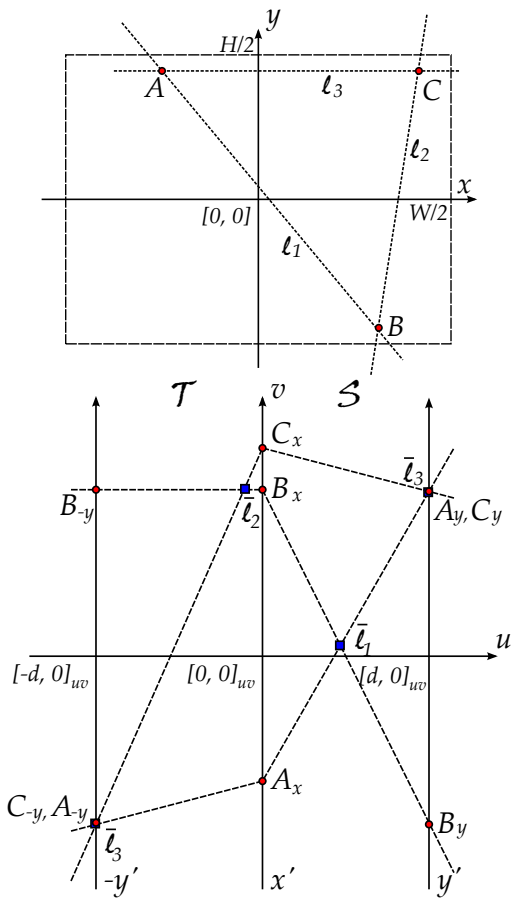

Figure 4. (top) Original x-y space and (bottom) its PClines representation – the corresponding $\mathcal{TS}$ space.

(top) shows the original x-y image with three points A, B, and C and three lines ℓ_1, ℓ_2, and ℓ_3 coincident with the points. The origin of x-y is placed into the middle of the image for convenience of the figures. Figure 4 (bottom) depicts the corresponding $\mathcal{TS}$ space. Note that a finite part of the u-v plane is sufficient:

$$\begin{aligned} -d &\le u \le d, \\ -\max\left(\frac{W}{2}, \frac{H}{2}\right) &\le v \le \max\left(\frac{W}{2}, \frac{H}{2}\right), \end{aligned} \tag{2}$$

where W and H are the width and height of the input raster image, respectively.

Any line $\ell : y = mx + b$ is now represented either by $\overline{\ell}_{\mathcal{S}}$ in the *straight* half or by $\overline{\ell}_{\mathcal{T}}$ in the *twisted* part of the u-v plane:

$$\begin{aligned} \overline{\ell}_{\mathcal{S}} &= \big(d, b, 1-m\big)_{\mathbb{P}^2}, -\infty \le m \le 0, \\ \overline{\ell}_{\mathcal{T}} &= \big(-d, -b, 1+m\big)_{\mathbb{P}^2}, 0 \le m \le \infty. \end{aligned} \tag{3}$$

Consequently, any line ℓ has exactly one image $\overline{\ell}$ in the $\mathcal{TS}$ space; except for cases that $m = 0$ and $m = \pm\infty$, when $\overline{\ell}$ lies in both spaces on y' or x', respectively. That allows the $\mathcal{T}$ and $\mathcal{S}$ spaces to be "attached" one to another. Figure 4 illustrates the spaces attached along the x' axis. Attaching also the y' and $-y'$ axes results in an enclosed Möbius strip.

To detect the lines, the standard Hough transform procedure is performed: the u-v space bounded by (2) is uniformly discretized into a matrix of accumulators; the input image is processed; and for all (or a selected subset) above-threshold pixels, a subset of the accumulators are incremented. In the case of PClines, two lines are rasterized for each input pixel: one in the straight half, one in the twisted half of the $\mathcal{TS}$ space. Horizontal coordinates of the lines' endpoints are fixed $\{0, d, -d\}$; vertical coordinates are directly the x, y and $-y$ coordinates of the original point.

Please note that for lines detected in the PClines space, the intersections with the x-y image boundary are easily computed in the following way. The $\mathcal{TS}$ space can be divided into convex parts by four lines:

$$v = \pm\frac{1}{2}\left(\pm\frac{W+H}{d}u + W\right). \tag{4}$$

According to the affiliation of a detected maximum $\overline{\ell}$ to one of these parts, the edges of the input image intersected by line ℓ can be determined and the intersections with them efficiently computed using the properties of parallel coordinates.

3.1. Resolution and Accuracy of the Algorithm

For inspection of error in localization of a line, line's parameters can be selected arbitrarily, we will be considering the differences in θ and in ϱ. The line localization error of the Hough transform can be predicted from the discretization density of the accumulator space. In the case of PClines, the density is proportional to the slope of functions $\theta(u)$ and $\varrho(u, v)$; higher slope means higher impact of

the changes in u and v on the values of θ and ϱ and thus higher discretization error. The θ-ϱ line parameterization [5] can be viewed as ideal in this sense because the slope is constant, so the discretization error is distributed uniformly across the Hough space.

In the $\mathcal{S}$ space, function $\theta(u)$ has its maximal slope when $u = d/2$, that is $m = -1$ (situation is analogous in the $\mathcal{T}$ space):

$$\theta = \arctan\left(\frac{u}{d-u}\right), \tag{5}$$

$$\frac{\partial\theta}{\partial u} = \frac{d}{(d-u)^2+u^2}. \tag{6}$$

Parameter θ is not dependent on v, so the discretization in the v axis has no influence on its error.

The ϱ parameter is a function of both u and v, as expressed in (7). For a given value of u, the discretization of $\varrho(u,v)$ is uniform across all v – see Eq. (8).

$$\varrho = \frac{vd}{\sqrt{(d-u)^2+u^2}}, \tag{7}$$

$$\frac{\partial\varrho}{\partial v} = \frac{d}{\sqrt{(d-u)^2+u^2}} \tag{8}$$

The $\varrho(u,v)$ discretization error at any location in the $\mathcal{TS}$ space can therefore be expressed as

$$\frac{\partial^2\varrho}{\partial u\partial v} = \frac{d(d-2u)}{\sqrt{(d-u)^2+u^2}^3}. \tag{9}$$

For convenience, two discretization error components ε_θ and ε_ϱ can be combined into one compound error $\varepsilon = \sqrt{\omega_\theta\varepsilon_\theta^2 + \omega_\varrho\varepsilon_\varrho^2}$ with weights ω_θ and ω_ϱ; this metric is used in the experimental evaluation in Sec. 4. Figure 5 contains a visualization of equations (6) and (9) (green, blue), and several examples of the combined error (red, gray). One combination is marked with red color as the "natural" one: $\omega_\theta = \omega_\varrho = 1$, discretization $\Delta u = \Delta v = 1$, $d = 384$.

However, also other factors, different from the discretization of the Hough space, influence its accuracy. For example, the typical way of rasterizing the sinusoids in the θ-ϱ parameterization is incrementing one accumulator for one θ value, which results in a different density of rasterized accumulators in the Hough space, causing additional errors; these errors are not as predictable as the discretization errors.

3.2. Detecting Parallel Lines and Vanishing Points

Because a line in the space of parallel coordinates represents one point in the original x-y space, images of lines $\overline{\ell}_1,\ldots,\overline{\ell}_n$ lie on one line if and only if lines $\ell_1,\ldots,\ell_n$ intersect in one point [12]. When $\overline{\ell}_1,\ldots,\overline{\ell}_n$ share the same u coordinate (their images lie on a vertical line in the PC space), the point of intersection is an ideal point (i.e. the lines are mutually parallel).

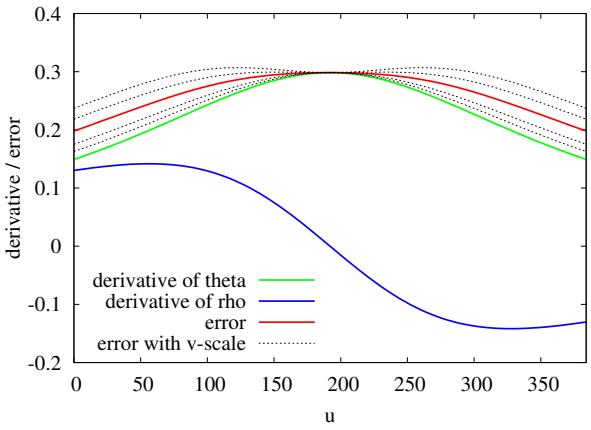

Figure 5. Dependency of derivatives and discretization errors on u in $\mathcal{S}$ and $\mathcal{T}$ spaces. Green: First-order partial derivative of θ with respect to u (6). Blue: Second-order mixed derivatives of ϱ with respect to u and v (9). Red: Combined error ε calculated from the derivatives. Dotted: Combined error calculated from derivatives with different discretization Δv.

Lutton et al. [13] determined vanishing points of 3D scenes in the θ-ϱ Hough space. Similar work is much more straightforward and efficient in the PClines: a RANSAC verifying hypotheses about collinearity of the found maxima. For detecting lines coincident with one point whose PC images lie both in the $\mathcal{S}$ and $\mathcal{T}$ spaces, a linear transformation from $\mathcal{S}$ to $\mathcal{T}$ and back exists:

$$\begin{aligned}\forall\ell : \overline{\ell}_\mathcal{S} = \big(u,v,1\big)_{\mathbb{P}^2} &\Rightarrow \overline{\ell}_\mathcal{T} = \big(u, v, \frac{d-2u}{d}\big)_{\mathbb{P}^2},\\ \forall\ell : \overline{\ell}_\mathcal{T} = \big(u,v,1\big)_{\mathbb{P}^2} &\Rightarrow \overline{\ell}_\mathcal{S} = \big(u, v, \frac{d+2u}{d}\big)_{\mathbb{P}^2}.\end{aligned} \tag{10}$$

Rapid detection of parallel lines can be used in the detection of barcodes, alignment of scanned text, etc. Detection of vanishing points and sets of lines which intersect in a common point have interesting applications in 3D reconstruction and augmented reality.

3.3. Shapes of the Maxima Detected in the Hough Space

Hough transform directly allows detecting infinite lines described by two parameters. However, Furukawa and Shinagawa [9] studied the possibility of also determining the extents of abscisses in the input image by analyzing the "butterfly" shapes around the maxima in the space of accumulators. They used the θ-ϱ parameterization as the most popular one. Using the parallel coordinates for this purpose may facilitate such processing because the boundaries of the "butterfly" are straight (the same holds for any PTLM-based Hough transform) – see Fig. 6. Close exploration of the possibilities of detecting actual line segments in an image by PClines is outside the scope of this paper.

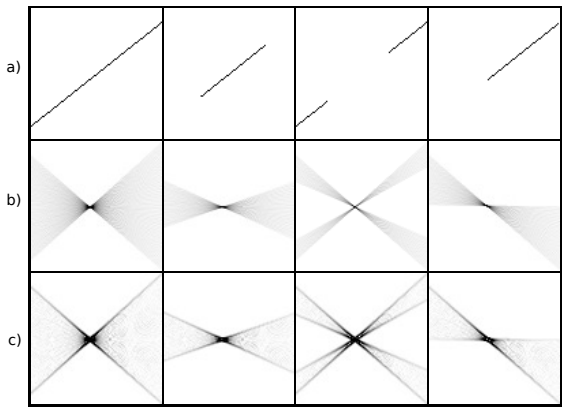

Figure 6. The "butterfly" shapes of the maxima in the PClines space. Columns: different segment of the same line. Rows: a) image of line segment. b) 96×96 pixels around the maxima in the PClines space, c) edges detected by Sobel operator.

3.4. Implementation Issues

Currently a very popular way of detecting lines in raster images is by using the Hough transform with the θ-ϱ parameterization (for example, OpenCV implements several variants of this parameterization and none other). However, the θ-ϱ line parameterization requires an evaluation of goniometric functions (which can be pre-computed in a fixed table) and multiplication of floating-point numbers. Several independent research groups have invested effort (e.g. [3], [7]) in coping with these requirements which are not severe on processors of personal computers, but they complicate any implementation by special processors or circuitry.

On the other hand, our parameterization PClines is based on rasterization of line segments, whose endpoints are defined by constants or by extremely simple functions of the point coordinates (e.g. $-y$). PClines can be easily accelerated by modern graphics chips. Modern GPUs are able to process a high number of points and at the same time accumulate to the Hough space of very high resolutions. Description of the implementaion or its experimental evaluation exceeds the scope of this paper. However, we are about to publish a detailed description of the implementation and its evaluation on real-life images. The results show that even for high-resolution ($\sim 3000 \times 2000$) real-life photographs, the detection using PClines typically takes less than 10 ms, only in difficult cases (many edge points, many lines) up to 20 ms.

Also, the rasterization of lines can be implemented in simple circuitry (e.g. Bresenham's algorithm), such as FPGA or ASIC. Simple signal processing chips and smart cameras can use PClines for fast and power-efficient line detection.

4. Experimental Evaluation

This section evaluates the accuracy of PClines and compares it to two existing parameterizations: to the θ-ϱ parameterization [5] and the original Hough's variant [10] denoted as the *slope-intercept* or m-b (line is $y = mx + b$) modified in order to be using two bounded spaces. The latter was selected as the dominant point-to-line mapping used in the context of the Hough transform.

In this evaluation, we used automatically generated data (similar as in [11]). The generator generates a black-and-white image (sized $W \times H$) by first rasterizing L lines directly from the line equation in its normal form; 8-connected neighborhood of pixels is used. Then, P noise pixel positions are randomly generated $p_i \in \{0, \ldots, W-1\} \times \{0, \ldots, H-1\}$, and the corresponding pixels are inverted in the image. Use of real-life images for accuracy evaluation is not feasible due to the lack of accurate-enough ground truth data.

Generated images are processed by the detectors (PClines, θ-ϱ, and m-b) and two errors of the detections are evaluated: ε_θ and ε_ϱ which are the differences from the ground truth in degrees or pixels, respectively. To obtain one error metric, a combined error $\varepsilon = \sqrt{\omega_\theta \varepsilon_\theta^2 + \omega_\varrho \varepsilon_\varrho^2}$ is used, with weights $\omega_\theta = \omega_\varrho = 1$. The accumulator space had the same dimensions for all three methods: for 512×512 images, accumulator space 768×724 was used.

The accuracy of a particular line parameterization for the Hough transform depends mainly on the discretization of the space of accumulators (Sec. 3.1). Figure 7 compares the errors of the three methods for lines with different θ generated within 5° wide intervals. For every interval of slopes, 100 images were generated, each with 1 line and $P = 25\,000$ noise pixels. We computed the average error of all lines detected in the images (Fig. 7 left) and the average of the 5 least accurate lines out of all 100 lines for each interval of slopes. The latter was computed to get a pessimistic estimation of the errors.

The measurements confirm the theoretical considerations: the θ-ϱ parameterization discretizes the space evenly; PClines are about as accurate as θ-ϱ for $\theta \in \{45^\circ, 135^\circ, 225^\circ, 315^\circ\}$ and more accurate at $\theta \in \{0^\circ, 90^\circ, 180^\circ, 270^\circ\}$; the m-b parameterization is the least accurate of the three evaluated methods.

5. Conclusions

This paper presents PClines – a new parameterization of lines for the Hough transform. The parameterization is computationally extremely efficient and the measurements show that it outperforms the commonly used variants of the Hough transform in accuracy, as well. PClines seem suitable for real-time line detection in simple hardware with low power consumption. Also, acceleration of PClines by

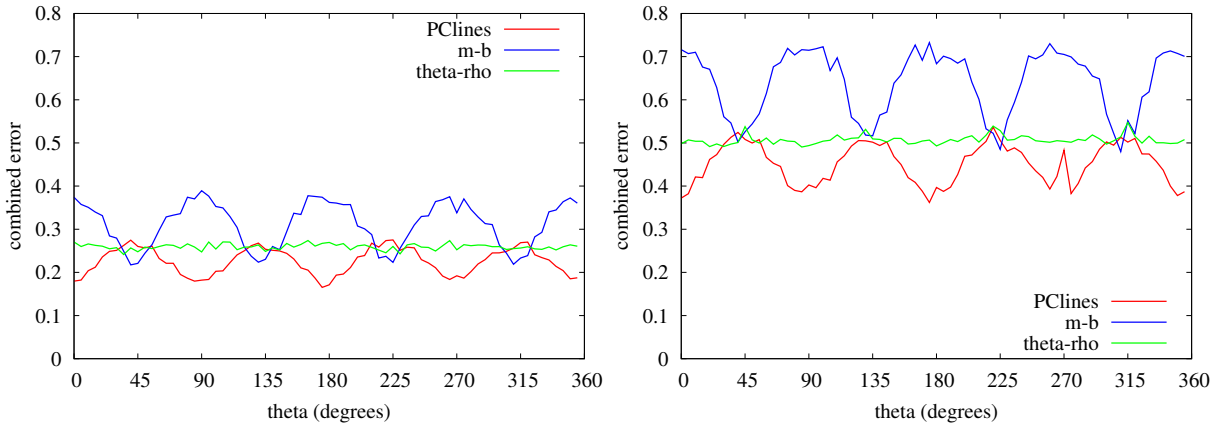

Figure 7. Line localization error as it depends on the lines' slope. For x on the horizontal scale, the lines' slope in degrees is at interval $\langle x, x+5)$. Red: PClines; Green: θ-ϱ; Blue: m-b. Left: average error over all lines; right: average error of the 5 least accurate lines, i.e. a pessimistic error estimation.

GPUs is straightforward and will be helpful in real-time image processing.

PClines allow convenient detection of parallel lines and lines coincident with one point. This opens space for future research, for example in 3D camera orientation estimation based on chessboard patterns and planar marks such as the squares used by the ARToolKit.

A generalization of our PClines parameterization can be used for detecting higher-dimensional linear objects: lines in 3D, planes in 3D, hyperplanes in multidimensional spaces, etc. Also, we intend to explore the possibility of using parallel coordinates for detection of other than linear objects. Simple geometric shapes, such as circles and conic sections, have well-described images in the space of parallel coordinates based on boundary contours [12].

Acknowledgements

This research was supported by the EU FP7-ARTEMIS project no. 100230 SMECY, by the research project CEZMSMT, MSM0021630528, and by the BUT FIT grant FIT-10-S-2.

References

[1] D. H. Ballard. Generalizing the Hough transform to detect arbitrary shapes. pages 714–725, 1987.

[2] P. Bhattacharya, A. Rosenfeld, and I. Weiss. Point-to-line mappings as Hough transforms. *Pattern Recognition Letters*, 23(14):1705–1710, 2002.

[3] J. D. Bruguera, N. Guil, T. Lang, J. Villalba, and E. L. Zapata. Cordic based parallel/pipelined architecture for the Hough transform. *J. VLSI Signal Process. Syst.*, 12(3):207–221, 1996.

[4] M. d'Ocagne. *Coordonnées parallèles et axiales. Méthode de transformation géométrique et procédé nouveau de calcul graphique déduits de la considération des coordonnées parallèlles*. Gauthier-Villars, 1885.

[5] R. O. Duda and P. E. Hart. Use of the Hough transformation to detect lines and curves in pictures. *Commun. ACM*, 15(1):11–15, 1972.

[6] U. Eckhardt and G. Maderlechner. Application of the projected Hough transform in picture processing. In *Proceedings of the 4th International Conference on Pattern Recognition*, pages 370–379, London, UK, 1988. Springer-Verlag.

[7] A. Fisher and P. Highnam. Computing the Hough transform on a scan line array processor (image processing). *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11:262–265, 1989.

[8] A. V. Forman. A modified Hough transform for detecting lines in digital imagery. In *Applications of artificial intelligence III*, pages 151–160, 1986.

[9] Y. Furukawa and Y. Shinagawa. Accurate and robust line segment extraction by analyzing distribution around peaks in Hough space. *Comput. Vis. Image Underst.*, 92(1):1–25, 2003.

[10] P. V. C. Hough. Method and means for recognizing complex patterns, Dec 1962. U.S. Patent 3,069,654.

[11] J. Illingworth and J. Kittler. The adaptive hough transform. *IEEE Trans. Pattern Anal. Mach. Intell.*, 9(5):690–698, 1987.

[12] A. Inselberg. *Parallel Coordinates; Visual Multidimensional Geometry and Its Applications*. Springer, 2009. ISBN: 978-0-387-21507-5.

[13] E. Lutton, H. Maitre, and J. Lopez-Krahe. Contribution to the determination of vanishing points using Hough transform. *IEEE Trans. Pattern Anal. Mach. Intell.*, 16(4):430–438, 1994.

[14] F. Natterer. *The mathematics of computerized tomography*. Wiley, John & Sons, Incorporated, 1986. ISBN 9780471909590.

[15] J. Princen, J. Illingowrth, and J. Kittler. Hypothesis testing: A framework for analyzing and optimizing Hough transform performance. *IEEE Trans. Pattern Anal. Mach. Intell.*, 16(4):329–341, 1994.

[16] R. Wallace. A modified Hough transform for lines. In *Proceedings of CVPR 1985*, pages 665–667, 1985.