Pattern Recognition 46 (2013) 725–740

Contents lists available at SciVerse ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

EDCircles: A real-time circle detector with a false detection control

Cuneyt Akinlar*, Cihan Topal

Department of Computer Engineering, Anadolu University, Eskisehir 26470, Turkey

ARTICLE INFO

Article history:
Received 9 April 2012
Received in revised form
21 September 2012
Accepted 26 September 2012
Available online 3 October 2012

Keywords:
Circle detection
Ellipse detection
Real-time image processing
Helmholtz Principle
NFA

ABSTRACT

We propose a real-time, parameter-free circle detection algorithm that has high detection rates, produces accurate results and controls the number of false circle detections. The algorithm makes use of the contiguous (connected) set of edge segments produced by our parameter-free edge segment detector, the Edge Drawing Parameter Free (EDPF) algorithm; hence the name EDCircles. The proposed algorithm first computes the edge segments in a given image using EDPF, which are then converted into line segments. The detected line segments are converted into circular arcs, which are joined together using two heuristic algorithms to detect candidate circles and near-circular ellipses. The candidates are finally validated by an *a contrario* validation step due to the Helmholtz principle, which eliminates false detections leaving only valid circles and near-circular ellipses. We show through experimentation that EDCircles works real-time (10–20 ms for 640 × 480 images), has high detection rates, produces accurate results, and is very suitable for the next generation real-time vision applications including automatic inspection of manufactured products, eye pupil detection, circular traffic sign detection, etc.

1. Introduction

Detection of circular objects in digital images is an important and recurring problem in image processing [1] and computer vision [2], and has many applications especially in such automation problems as automatic inspection of manufactured products [3], aided vectorization of line drawing images [4,5], pupil and iris detection [6–8], circular traffic sign detection [9–11], and many others.

An ideal circle detection algorithm should run with a fixed set of internal parameters for all images, i.e., require no parameter tuning for different images, be very fast (real-time if possible), detect multiple small and large circles, work with synthetic, natural and noisy images, have high detection rate and good accuracy, and produce a few or no false detections. The circle detection algorithm presented in this paper satisfies all of these properties.

Traditionally, the most popular circle detection techniques are based on the famous circle Hough transform (CHT) [12–16]. These techniques first compute an edge map of the image using a traditional edge detector such as Canny [17], map the edge pixels into the three dimensional Hough circle space (x, y, r) and extract circles that contain a certain number of edge pixels. Not only CHT-based techniques are very slow and memory-demanding, but they also produce many false detections especially in the presence of noise. Additionally, these methods have many parameters that must be preset by the user, which greatly limits their use.

To overcome the limitations of the classical CHT-based methods, many variants have been proposed including probabilistic HT [18,19], randomized HT [20,21], fuzzy HT [22], etc. There are also approaches based on HT and hypothesis filtering [23–25]. All these methods try to correct different shortcomings of CHT, but are still memory-demanding and slow to be of any use in real-time applications.

Apart from the CHT-based methods, there are several randomized algorithms for circle detection. Chen et al. [26] propose a randomized circle detection (RCD) algorithm that randomly selects four pixels from the edge map of an image, uses a distance criterion to determine whether there is a possible circle in the image. They then use an evidence-collecting step to test if the candidate circle is a real-circle. RCD produces good results, but is slow. Recently, Chung et al. [27,28] have proposed efficient sampling and refinement strategies to speed up RCD and increase the accuracy of RCD's results. Although the new RCD variants named GRCD-R, GLRCD-R [28] have good detection rates and produce accurate results, they still are far from being real-time. Furthermore, all RCD-variants work on the edge map of an image computed by a traditional edge detector such as the Sobel filter or the Canny edge detector, which have many parameters that must be set by the user.

Recently, many efforts have concentrated on using genetic algorithms and evolutionary computation techniques in circle detection [29–36]. Ayala-Ramirez et al. [30] proposed a genetic algorithm (GA) for circle detection, which is capable of detecting multiple circles but fails frequently to detect small or imperfect

* Corresponding author. Tel.: +90 2223213550x6553.
E-mail addresses: cakinlar@anadolu.edu.tr (C. Akinlar), cihant@andolu.edu.tr (C. Topal).

0031-3203/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.patcog.2012.09.020

circles. Dasgupta et al. [31–33] developed a swarm intelligence technique named adaptive bacterial foraging optimization (ABFO) for circle detection. Their algorithm produces good results but is sensitive to noise. Cuevas et al. use discrete differential evolution (DDE) optimization [34], harmony search optimization (HSA) [35] and an artificial immune system optimization technique named Clonal Selection Algorithm (CSA) [36] for circle detection. Although these evolutionary computation techniques have good detection rates and accurate results, they usually require multiple runs to detect multiple circles, and are quite slow to be suitable for real-time applications. Just like RCD, these algorithms work on an edge map pre-computed by a traditional edge detection algorithm with many parameters.

Frosio et al. [37] propose a real-time circle detection algorithm based on maximum likelihood. Their method is fast and can detect partially occluded circular objects, but requires that the radius of the circles to be detected be predefined, which greatly limits its applications. Wu et al. [41] present a circle detection algorithm that runs 7 frames/s on 640×480 images. The authors claim to achieve high success rate, but there is not much experimental validation to back their claims. Zhang et al. [38] propose an ellipse detection algorithm that can be used for real-time face detection. Liu et al. [39] present an ellipse detector for noisy images and Prasad et al. [40] present an ellipse detector using the edge curvature and convexity information. While both algorithms produce good results, they are slow and not suitable for real-time applications.

Vizireanu et al. [42–44] make use of mathematical morphology for shape decomposition of an image and use the morphological shape decomposition representation of the image for recognition of different shapes and patterns in the image. While their algorithms are good for the detection of general shapes in an image, they are not suitable for real-time applications.

Desolneux et al. [60] is the first to talk about the *a contrario* circular arc detection. Recently, Patraucean et al. [45,46] propose a parameter-free ellipse detection algorithm based on the *a contrario* framework of Desolneux et al. [58]. The authors extend the line segment detector (LSD) by Grompone von Gioi et al. [63] to detect circular and elliptic arcs in a given image without requiring any parameters, while controlling the number of false detections by the Helmholtz principle [58]. They then use the proposed algorithm (named ELSD [46]) for the detection and identification of Bubble Tags [47].

In this paper, we present a real-time (10–20 ms on 640×480 images), parameter-free circle detection algorithm that has high detection rates, produces accurate results, and has an *a contrario* validation step due to the Helmholtz principle that lets it control the number of false detections. The proposed algorithm makes use of the contiguous (connected) set of edge segments produced by our parameter-free edge segment detector, the edge drawing parameter free (EDPF) [48–53]; hence the name EDCircles [54,55]. Given an input image, EDCircles first computes the edge segments of the image using EDPF. Next, the resulting edge segments are turned into line segments using our line segment detector, EDLines [56,57]. Computed lines are then converted into arcs, which are combined together using two heuristic algorithms to generate many candidate circles and near-circular ellipses. Finally, the candidates are validated by the Helmholtz principle [58–63], which eliminates false detections leaving only valid circles and near-circular ellipses.

2. The proposed algorithm: EDCircles

EDCircles follows several steps to compute the circles in a given image. The general idea is to extract line segments in an image, convert them into circular arcs and then combine these arcs to detect circles and near-circular ellipses. General outline of EDCircles algorithm is presented in Algorithm 1 and we will describe each step of EDCircles in detail in the following sections.

Algorithm 1. Steps of EDCircles algorithm.

1. Detect edge segments by EDPF and extract complete circles and ellipses.
2. Convert the remaining edge segments into line segments.
3. Detect arcs by combining line segments.
4. Join arcs to detect circle candidates.
5. Join the remaining arcs to detect near-circular ellipse candidates.
6. Validate the candidate circles/ellipses using the Helmholtz principle.
7. Output the remaining valid circles/ellipses.

2.1. Edge segment detection by edge drawing parameter free (EDPF)

Given an image, the first step of EDCircles is the detection of the edge segments in the image. To achieve this, we employ our recently proposed, real-time edge/edge segment detector, edge drawing (ED) [48–51]. Unlike traditional edge detectors, e.g., Canny [17], which work by identifying a set of potential edge pixels in an image and eliminating non-edge pixels through operations such as non-maximal suppression, hysteresis thresholding, erosion, etc., ED follows a proactive approach and works by first identifying a set of points in the image, called the anchors, and then joins these anchors using a smart routing procedure; that is, ED literally draws edges in an image. ED outputs not only a binary edge map similar to those output by traditional edge detectors, but it also outputs the result as a set of edge segments each of which is a contiguous (connected) pixel chain [49].

ED has many parameters that must be set by the user, which requires the tuning of ED's parameters for different types of images. Ideally, one would want to have a real-time edge/edge segment detector which runs with a fixed set of internal parameters for all types of images and requires no parameter tuning. To achieve this goal, we have recently incorporated ED with the *a contrario* edge validation mechanism due to the Helmholtz principle [58–60], and obtained a real-time parameter-free edge segment detector, which we name edge drawing parameter free (EDPF) [52,53]. EDPF works by running ED with all ED's parameters at their extremes, which detects all possible edge segments in a given image with many false positives. We then validate the extracted edge segments by the Helmholtz principle, which eliminates false detections leaving only perceptually meaningful edge segments with respect to the *a contrario* approach.

Fig. 1(a) shows a 424×436 grayscale synthetic image containing a big circle obstructed by four rectangular blocks, a small ellipse obstructed by three rectangular blocks, a small circle, an ellipse and an arbitrary polygon-like object. When this image is fed into EDPF, the edge segments shown in Fig. 1(b) are produced. Each color in the edge map represents a different edge segment, each of which is a contiguous chain of pixels. For this image, EDPF outputs 15 edge segments in just 3.7 ms in a PC with 2.2 GHz Intel 2670QM CPU. Notice the high quality nature of the edge map with all details clearly visible.

Each edge segment traces the boundary of one or more objects in the figure. While the boundary of an object may be traced by a single edge segment, as the small circle, the ellipse and the polygonal object are in Fig. 1(b), it is also possible that an object's boundary be traced by many different edge segments. This is the case for the big circle as the circle's boundary is traced by four different edge segments, and the small obstructed ellipse, which is traced by three different edge segments. The result totally depends on the structure of the objects, the amount of obstruction and noise in the image. That is, there is no