

# Polygon Approximations of the Euclidean Circles on the Square Grid by Broadcasting Sequences

Haomin Song and Igor Potapov

University of Liverpool, Liverpool, United Kingdom

H.Song12@student.liverpool.ac.uk; Potapov@liverpool.ac.uk

**Abstract.** Euclidean circle approximation on the square grid is an important problem in digital geometry. Recently several schemes have been proposed for approximation of Euclidean circles based on Neighbourhood Sequences, which correspond to repeated application of the von Neumann and Moore neighbourhoods on a square grid. In this paper we study polygon approximations of the Euclidean circles on the square grid with Broadcasting Sequences which can be seen as a generalization of Neighbourhood Sequences. The polygons generated by Broadcasting Sequences are the Minkowski sums of digital disks defined by a given set of broadcasting radii. We propose a polynomial time algorithm that can generate Broadcasting Sequences which are providing flexible and accurate approximation of Euclidean circles.

## 1 Introduction

In the area of digital image processing the Distance Transform (DT) computation is computationally expensive if it is based on explicit calculation of Euclidean distances. In order to overcome this problem, other distance metrics (such as Manhattan, Chessboard, Neighbourhood, Chamfer, Broadcasting) have been proposed to calculate the distance considering only small neighbourhoods. The transforms that apply these metrics can be seen as an approximation of Euclidean distance with considerably less complex algorithms and which is suitable for parallel implementation on cellular processor arrays. In this context approximation of Euclidean distance leads to the problem of Euclidean circle approximation on regular grids [1–3]. Polygonal approximation is one of the efficient approaches to approximate a circle in a discrete domain [4]. The main objective is to approximate a given circle by a polygonal chain satisfying certain optimality criteria, such as minimising global approximation or restricting local error in some predefined threshold. Moreover, such approximations usually require to reduce storage spaces that needed to represent a shape of some circle as well as to improve the extraction and properties of required features from a given set of digital curves [5]. The simpler and more efficient approximations has been constructed then more likely it can fit to a larger class of applied problems.

Several schemes have been proposed for an approximation of circles based on Neighbourhood Sequences, which is an application of the von Neumann and Moore neighbourhoods on the square lattice [6–8], triangular [9] or hexagon

lattices [2]. In case of the neighbourhood sequences the best approximation for a circle on the square lattice is the regular octagon and the problem of generating such shape by neighbourhood sequences has been analysed in [10]. The main downsides for a circle approximation by this approach are high error rate and low expansibility due to restricted polygonal shapes.

This paper provides another methodology, using broadcasting automata as a tool, to implement polygonal approximation. Broadcasting Automata can be defined on some form of grid or lattice structure and have a simple computational primitives comparative to finite state automata with the ability to receive and send messages both from and to those automata which are within its transmission radius [11]. In this work, we only consider the square grid structure, but due to flexibility of the model similar broadcasting processes can be defined on other lattices or other distance measured methods. However in connection with simple agregation functions it can be used to generate more complex non-convex shapes, the approximation of  $L^p$  metric or even arbitrary patterns, see [11].

Originally the broadcasting automata has been studied in the context of the distributed algorithms and pattern formation problems. Each state in the grid or lattice is regarded as a cell, which only has limited sensor range and restricted common knowledge such as positions [12]. Therefore, all those cells within a certain transmission radius, receive the message from the sender. As a result, the communication between cells works like a Multi Agent System, which increases the robustness and ability to tackle failures and abnormal conditions [13]. In analogy to the neighbourhood sequences the similar broadcasting sequences can be defined and in the case of 2D grid it covers classical Moore and Von Neumann neighbourhoods as well as other neighbourhoods corresponding to digital disks.

By changing broadcasting radii we are changing sizes and shapes of transmission polygons on a grid (i.e. digital disks) [12]. Obviously using larger number of radii corresponds to transmission polygons with larger number of line segments which in its turn leads to better approximation of the circles. So along this line we are focusing on the problem of finding the best approximation of the Euclidean circles for a given fixed set of transmission radii. Following recent results from [12] about the composition properties of their Minkowski sum and methods for efficient description of the polygons by the chain codes we design an algorithm to find a Broadcasting Sequence that provides polygon approximation of Euclidean circles. The error rate between polygon and circle has been used as criterion to evaluate efficiency. We provide the analysis of the time complexity for the proposed algorithm and experimental results to compare generated approximations with optimal solution for a given set of radii and the distance to the Euclidean circles.

## 2 Approximation with neighbourhood sequences

The circle approximation using neighbourhood sequences on a square grid has been studied in [10]. The neighbourhood sequences can generate octagon and

the more regular octagon can be generated the better approximation can be achieved. The area of a polygon is defined by the Equation (1) in [10]

$$S = a^2 + 2\sqrt{2}ab + b^2, \tag{1}$$

where  $a$  and  $b$  are side-lengths of the sequence-generated octagon. So the best approximation can be achieved when the side-lengths are the same, i.e. when  $a = b$ . In this case, the area of the ideal polygon is equal to  $S_{polygon} = (2 + 2\sqrt{2})a^2$ . The relationship between  $a$  and the radius of circumscribed circle  $R$  is represented by the equation  $a = \sqrt{2 - \sqrt{2}}R$ . In that case, the area of the circumscribed circle is  $S_{circumscribed} = \pi R^2 = (1 + \frac{\sqrt{2}}{2})\pi a^2$ . To evaluate the approximation between polygon and the circumscribed circle the following Error Rate (ER) defined by Equation (2) has been used in [10]

$$ER = 1 - \frac{S_{polygon}}{S_{circumscribed}}. \tag{2}$$

It is obvious that the better approximation corresponds to lower error rate and the error rate of neighbourhood sequences approximation is about 9.97% as shown in [10]. The objective for this paper is to provide more accurate polygonal approximation and to design an algorithm for its construction.

### 3 Broadcasting sequences

As the alternative to the neighbourhood sequences we consider the problem of approximation of Euclidean circles by polygons generated by a set of neighbourhoods which correspond to a finite set of digital disks. Such sequences are known as *Broadcasting sequences* [11]. They are naturally generalizing neighbourhood sequences and enriching the model by a variety of generated polygons. However finding the optimal approximation of Euclidean circles by broadcasting sequences for a given finite set of digital disks is quite nontrivial computational problem, since it is hard to provide simple canonical forms corresponding to the optimal solutions for a set of digital disks.

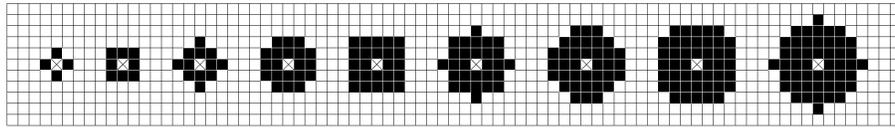
In this section we introduce basic concepts of broadcasting sequences and digitized circles, which will be used as neighbourhoods for approximation of Euclidean circles. Originally broadcasting automata model has been defined in [11, 12] as a network of finite automata on the integer grid. The mechanism for communication between automata is message passing. The messages generated by an automaton (sender) transfer to its transmission neighbourhood (receiver), which is a subset of its neighbors within the transmission range. Messages are generated and transferred instantaneously at a discrete time step, regarding as synchronous steps [12]. In this paper we study the concept of transmission neighbourhoods in the context of the Euclidean circle approximation problem.

**Definition 1** *The **Euclidean distance** between two cells  $c_1$  and  $c_2$  on a square grid with integer coordinates  $(x_1, y_1)$  and  $(x_2, y_2)$  is defined as*

$$d(c_1, c_2) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}.$$

**Definition 2** *The square transmission radius  $r$ , which is used to quantify the transmission range, is the square of the max cell distance of the sender and its receivers. The Broadcasting Neighbourhood of the cell  $c_0$  on the distance  $\sqrt{r}$  is defined as a set  $\{c | d(c, c_0)^2 \leq r\}$  which we also call as **primitive transmission polygon (PTP)**.*

Varying the transmission radius we construct different digital disks, i.e. different sizes and shapes of PTP. We also use a notion of the square radius to identify a PTP for simplicity to stay with integer numbers. Let us illustrate a variety of PTPs corresponding to different square transmission radii on Figure 1 [11].



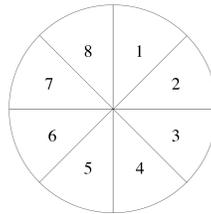
**Fig. 1.** Examples of PTPs with square transmission radii in order  $\mathcal{R} = \{1, 2, 4, 5, 8, 9, 10, 13, 16\}$

**Definition 3** *A Broadcasting sequence is a sequence of radii*

$$\mathcal{R} = \{r_1, r_2, r_3 \dots r_n\},$$

where  $r_i$  is a square transmission radius and  $n$  is the number of elements in the sequence. The polygon generated by broadcasting sequence is the Minkowski sum of all primitive transmission polygons (digital disks) with radii from  $\mathcal{R}$ .<sup>1</sup>

In order to represent symmetrical polygons generated by broadcasting sequences we use the chain code [14]. The polygon is divided into eight octants (Figure 2) and it is sufficient and necessary to depict the first octant of a polygon because the rest of the part is mirrored from the first octant [11].



**Fig. 2.** Eight octants of digital polygons

<sup>1</sup> Minkowski sum gives the same polygon for different order of the composition as it is commutative operation.

**Definition 4** [11] A **chain code** for a digitized circle in the first octant is a word in  $\{0, 1\}^*$  where 0 is positive motion along the x-axis, 1 is positive motion along the x-axis and negative motion along the y-axis. The 9<sup>th</sup> image in Figure 1 is an example of chain code 101.

It has been proven in [11] that the edges of primitive transmission polygon in the first octant can be represented by chain code the the formal definition of the line segments has been introduced to describe the sides of these polygons.

**Definition 5** [11] **Line segments**, sub-word of chain code, represent a straight edge of a circle in one of the following forms:  $0^*$ ,  $(10^n)^*$ ,  $(10^n)(10^{n+1})^*$ ,  $(10^n)^*(10^{n+1})$ . For example, the line segments of 9<sup>th</sup> polygon in Figure 1 are:  $\{10, 1\}$ .

**Definition 6** [11] A **gradient**,  $G(s)$ , of a line segment  $s$ , is  $G(s) = \frac{|s|_1}{|s|}$ , where the function  $|s|_1$  returns the number of 1s in the chain code and  $|s|$  is the length of  $s$ . It is also called tangent of line segment. For example, the gradient of line segment "1000" is 0.25.

It was shown in [11] that for these polygons, the gradients of line segments in the first octant are strictly increasing and the following composition theorem holds:

**Theorem 1** [11] **Composition theorem:** Given two chain codes  $u$  and  $v$  which contain line segment  $l_1^u l_2^u \dots l_t^u$  and  $l_1^v l_2^v \dots l_t^v$  with strictly increasing gradients, the chain code of a composition of  $u$  and  $v$  can be constructed by combining line segments of  $u$  and  $v$ , then ordering them by increasing gradient. The composition of two chain codes  $u$  and  $v$  is **commutative** and corresponds to the chain code of Minkowski sum of digital disks in the first octant.

**Definition 7** A **length**,  $L(s)$ , of a line segment  $s$ , is  $L(s) = |s| \sqrt{1 + G(s)^2}$ , where  $|s|$  is the length of line segment  $s$ ,  $G(s)$  is the gradient of  $s$ .

## 4 Polygon approximation of circle

In this paper we design the algorithm to find a polygon approximation of a circle using broadcasting sequences. Single primitive transmission polygons may have a very high error rate for circle approximation and clearly polygons with more segments can achieve better approximation rate circle. Following Theorem 1 it is clear that the composition of multiple PTPs contains more types of line segments, which results in a better approximation for circle. However when the total number of radii is fixed the main question is to find the right ratios of radii that should be used together for the best approximation of a circle.

The problem of finding the best **approximation of Euclidean circle** can be formulated as follows: Given a set of square transmission radii  $\{r_1, r_2, \dots, r_m\}$  and the maximal length of broadcasting sequence  $h$  find a broadcasting sequence

$$\mathcal{R} = \{r_1 \dots r_1, r_2 \dots r_2, \dots, r_m \dots r_m\},$$

where  $|\mathcal{R}| \leq h$  such that the polygon generated by a sequence  $\mathcal{R}$  (i.e. Minkowski sum of digital disks with radii from  $\mathcal{R}$ ) has the smaller error rate in relation to Euclidean circle.

In order to solve the above problem we represent polygons in the vector form, look for the ideal polygon based on the line segments and finally approximate the ideal polygon by given PTPs. So the algorithm contains three parts: vector system setup, ideal polygon and broadcasting sequence construction.

#### 4.1 Vector system setup

The vector system is used to represent polygons by line segments. Each line segment in a polygon has two associated values: the gradient and the length. These line segments can form a polygon by strict increasing gradients in the first octant. In that case, **gradient set** records gradients of line segments and **length vector** records lengths of corresponding line segments.

**Definition 8 Gradient set**,  $GV(p) = \{g_1, g_2, \dots, g_n\}$ , of the polygon  $p$ , denotes a list of gradients of line segments of  $p$ . For convenience, gradients in the set are in ascending order ( $g_i < g_{i+1}$ ).

**Definition 9 Length vector**,  $LV(p) = [l_1, l_2, \dots, l_n]$ , of the polygon  $p$ , denotes a list of length of line segments of  $p$ . The elements of the same index in  $GV(p)$  and  $LV(p)$  indicate the gradient and length of one line segment respectively.

In order to operate with several polygons we will introduce extensive length vector. It represents a length vector of  $p$  in the context of larger gradient set of  $p_0$ , where  $GV(p)$  is the subset of  $GV(p_0)$ .

**Definition 10 Extensive length vector**,  $LV(p, p_0) = [l_1, l_2, \dots, l_n]$ , of the polygon  $p$ , denotes a list of length of line segments of  $p$  corresponding to  $GV(p_0)$  and if some  $g$  exist in  $GV(p_0)$  but not in  $GV(p)$ , the corresponding element in this vector is 0. The size of this vector is the same as  $GV(p_0)$ .

Later we use  $p_0$  to represent the objective polygon and  $PTPs = \{p_1, p_2, \dots, p_m\}$  represents all PTPs according to  $\mathcal{R}$ . Since the objective polygon is generated by the given PTPs, gradient set of objective polygon is identified as  $GV(p_0) = \bigcup_{i=1}^m GV(p_i)$ . In that case, the gradient set of each PTP is the subset of  $GV(p_0)$ .

**Definition 11 Length matrix** of a list of  $PTPs = (p_1, p_2, \dots, p_m)$  is  $LM(PTPs, p_0) = [LV(p_1, p_0)^T, LV(p_2, p_0)^T, \dots, LV(p_m, p_0)^T]$ , which is a matrix of lengths of line segments of PTPs corresponding to  $GV(p_0)$ . The size of this matrix is the product of the size of PTPs and the size of  $GV(p_0)$ .

**Example:** Let us consider a set of PTPs  $P = \{p_1, p_2, p_3\}$  with corresponding radii  $\{r_1 = 16, r_2 = 18, r_3 = 25\}$ . Let  $p_0$  be the objective polygon, then the Gradient set of  $p_1, p_2, p_3$  and  $p_0$  will be defined as  $GV(p_1) = \{\frac{1}{2}, 1\}$ ,  $GV(p_2) = \{0, \frac{1}{2}\}$ ,  $GV(p_3) = \{\frac{1}{3}, 1\}$  and  $GV(p_0) = \{0, \frac{1}{3}, \frac{1}{2}, 1\}$ . The Extensive Length vectors of

PTPs in this case are:  $LV(p_1, p_0) = \{0, 0, \sqrt{5}, \sqrt{2}\}$ ,  $LV(p_2, p_0) = \{2, 0, \sqrt{5}, 0\}$  and  $LV(p_3, p_0) = \{0, \sqrt{10}, 0, \sqrt{2}\}$  and the Length matrix is:

$$LM(P, p_0) = \begin{bmatrix} 0 & 2 & 0 \\ 0 & 0 & \sqrt{10} \\ \sqrt{5} & \sqrt{5} & 0 \\ \sqrt{2} & 0 & \sqrt{2} \end{bmatrix}.$$

Thus the length matrix  $X = LM(PTPs, GV(p_0))$  represents PTPs and the length vector  $Y = LV(p_0)$  represents the length vector of objective polygon. We use the notion of the Parikh vector to combine these two objects  $X$  and  $Y$ .

**Definition 12** [15] Let  $\Sigma = \{a_1, a_2 \dots a_k\}$  be an alphabet. The **Parikh vector** of a word  $w$  is defined as the function  $p(w) = (|w|_{a_1}, |w|_{a_2}, |w|_{a_3} \dots |w|_{a_k})$ , where  $|w|_{a_i}$  denotes the number of occurrences of the letter  $a_i$  in the word  $w$ .

In other words, the elements in Parikh vector are the number of occurrences of corresponding radii that appear in the broadcasting sequence. Finally, the relationship of length vector of objective polygon  $Y$ , the length matrix of PTPs  $X$  and the Parikh vector  $P$  can be defined as Equation(3).

$$XP = Y \tag{3}$$

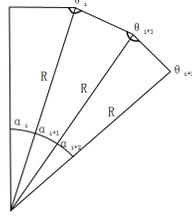
The value of  $X$  can be derived from PTPs, however the values of  $Y$  and  $P$  are unknown. In subsection 4.2 we show how to find the value of  $Y$  and in the subsection 4.3 we will calculate  $P$  from  $X$  and  $Y$ .

#### 4.2 Ideal polygon construction

Let us start with finding the length vector of the ideal polygon  $Y$  according to the gradient set  $GV(p_0)$ , which contains all gradients of line segments that PTPs provide. Since the polygon is an inscribed polygon with max area, each vertex of polygon should be on the circle with the same distance to the center of circle and the polygon occupies the circle as large proposition as possible. To calculate the area of polygon, the polygon is divided into isosceles triangles whose point angles are the center of the circle and the base edges are the edges of polygon. In that case, the area of polygon is the sum of areas of all triangles, see Figure 3. In addition, since the gradients of edges are fixed, there are constraints (Equation (4)) among point angles and base angles, i.e  $\theta_i$  and  $\alpha_i$ , where  $\theta_i$  is known as outer angle of polygon and  $\alpha_i$  is as the inner angle of a divided triangle of the polygon, both shown on Figure 3:

$$2\pi - 2\theta_i - \alpha_i - \alpha_{i+1} = 0 \tag{4}$$

Due to the symmetry of constructed polygons, we only need to count up triangles in the first octant, which occupies  $\frac{1}{8}$  of the total polygon. However, due to the double line segments, the triangles with gradient of 0 or 1 only half



**Fig. 3.** Triangle division of polygon

occupy the first octant. In that case, when calculating the area, the area should be half for these triangles. The area function of a polygon in the first octant can be defined as Equation (5) and the objective is to find the maximum of  $S_P$ .

$$S_P(\alpha) = \begin{cases} \frac{1}{2}R^2 \sum_1^n \sin \alpha_i & \text{no double line segment} \\ \frac{1}{2}R^2 (\sum_2^n \sin \alpha_i + \frac{1}{2} \sin \alpha_1) & G(\alpha_1) = 0, G(\alpha_n) \neq 1 \\ \frac{1}{2}R^2 (\sum_1^{n-1} \sin \alpha_i + \frac{1}{2} \sin \alpha_n) & G(\alpha_1) \neq 0, G(\alpha_n) = 1 \\ \frac{1}{2}R^2 (\sum_2^{n-1} \sin \alpha_i + \frac{1}{2}(\sin \alpha_1 + \sin \alpha_n)) & G(\alpha_1) = 0, G(\alpha_n) = 1 \end{cases} \quad (5)$$

From the criterion equation, since the circumscribed circle is fixed, it is clear that the larger area the polygon corresponds to the lower error rate of an approximation. Therefore, we need to find the maxima of this function subject to equality constraints. We use Lagrange Multiplier method, which is the proper mathematical optimization technique to find the solutions of the function. Therefore, for constraints, we define scalars  $\lambda_1, \lambda_2, \dots, \lambda_{n-1}$  and an auxiliary function shown in Equation (6), then we solve the Equation set (7). There are  $2n - 1$  equations and  $2n - 1$  variables, which are solvable.

$$F = S_P(\alpha) + \sum_{i=1}^{n-1} \lambda_i (2\pi - 2\theta_i - \alpha_i - \alpha_{i+1}) \quad (6)$$

$$F(\alpha; \lambda) = \begin{cases} \nabla S_P(\alpha) - \sum_{k=1}^{n-1} \lambda_k \nabla (2\pi - 2\theta_i - \alpha_i - \alpha_{i+1}) = 0 \\ 2\pi - 2\theta_i - \alpha_i - \alpha_{i+1} = 0 \end{cases} \quad (7)$$

Since the above equations are not linear, we utilized Newton-Raphson method to solve the equation set, which is the method of finding successively better approximations to the roots of a real-valued function. All variables are combined in a vector  $x = [\alpha; \lambda]$ . The iterative equation can be defined as Equation (8)

$$x^{(k+1)} = x^{(k)} - J^{-1}(x^{(k)})S_P(x^{(k)}) \quad (8)$$

where  $x^{(k)}$  is previous point,  $x^{(k+1)}$  is the derivative point,  $S_P$  is the column vector of equations at  $x^{(k)}$  and  $J$  is Jacobian matrix of  $S_P$  at  $x^{(k)}$ , a matrix of all first-order partial derivatives of  $S_P$ .

We create an iterative algorithm to implement this method. In the algorithm, we define the max iterative time  $max\_itr$  and deviation range  $err\_range$ . The algorithm repeats until deviation between calculated function and objective values is smaller than the deviation range. The founded  $x$  is result of equation set and  $k$  is total iterative step. If the iterative time overtakes the predefined value  $max\_itr$ , the algorithm returns no results.

Consequently, the resulted vector  $x = [\alpha; \lambda]$  of Newton-Raphson method contains the sine values of ideal angles of triangles in the first octant. Based on the equation  $l_i = 2R \sin \alpha_i$ , the ratio of lengths of line segment is the same as ratio of sine values of inner angles since that all divided triangles are isosceles with equal side length  $R$ . In addition, only the shape of polygon is considered. Therefore, the relative lengths of line segments should be adopted.

### 4.3 Broadcasting sequence construction

In this section we will show how to calculate  $P$  and find the broadcasting sequence which can be close to the ideal polygon. In the section 4.1 we show that the ideal result of Parikh vector should satisfies to the Equation 3, however, for length matrix  $X$ , we still need to consider several cases:

1. If the number of rows is smaller than the number of columns, which means the number of PTPs is larger than the number of gradient types, the result of  $P$  is not unique and infinite.
2. If the number of rows equal to the number of columns, which means the number of PTPs is the same as the number of gradient types, the result of  $P$  is unique.
3. If the number of rows is larger than the number of columns, which means the number of PTPs is larger than the number of gradient types, the result of  $P$  is not exist.

For the first case, we can define some elements of  $P$  in the admissible domain until the number of undefined elements is equal to the number of columns. Then the result of  $P$  is unique and computable. For the second case,  $P$  is derived directly by the Equation (9)

$$P = X^{-1}Y. \tag{9}$$

For the third case, assuming that there is no  $P$  that satisfy the equation, we aim to find the best fitting that minimizes the deviation. The Least Square Fitting is the most common method which solves the Equation (10)

$$S(P) = \arg \min(\|XP - Y\|)^2. \tag{10}$$

We also need to assume that all elements in  $P$  must be positive. In that case, a constraint is added in the equation and another extensive Least Square Fitting method— Non-negative least squares (NNLS) with the constraint  $x > 0$  for equation. This method is based on the observation that only a small subset

of constraints is usually active at the solution [16], which is given by Lawson and Hanson [17]. The basic principle uses iterative method to calculate the equation until Kuhn-Tucker conditions are satisfied. Currently, MathWorks [18] provides the function for the algorithm NNLS, which is called to “lsqnonneg”. Therefore, the Equation (11) can be used directly for a given  $X$  and  $Y$

$$P = (X^T X)^{-1} X^T Y. \quad (11)$$

After generating  $P$  we can derive the broadcasting sequence. In the broadcasting sequence, the number of radii is the values of corresponding elements in  $P$ . Note that the absolute values of elements in  $P$  are not important. The main focus are the relative values among elements. The values of elements in  $P$  can be shrunk by same proportion according to the accuracy requirements. Considering that the number of elements in broadcasting sequence should be integers, all elements in  $P$  shrink and approximate an integer within the requirement range.

**Proposition 1.** *For a given set of radii  $r_1, \dots, r_m$  with corresponding number of line segments  $n_1, \dots, n_m$  the constructed algorithm can find the broadcasting sequence approximating Euclidean circle in polynomial time  $O(nm^2 + n^3k)$  where  $n$  is the sum of all  $n_i$  for  $i = 1..m$  and assuming that  $k \propto \log t$  where  $t$  is approximation precision.*

*Proof.* According to the section 4, which describes the whole process of broadcasting sequence construction, there are 3 steps needed to be considered.

In the first step “Vector system setup”, based on the given set of square radii where the size of set is  $m$  and max value of square radius is  $r_{max}$ , the max number of line segments  $n$  for each PTP satisfies the equation  $n$ , which is a linear relationship to square root of  $r_{max}$ :  $n \propto \sqrt{r_{max}}$ . In this case, the time complexity for deriving the set of line segments of objective polygon  $p_0$  is  $O(nm)$ . According to the definition (10), the time complexity to derive the length matrix  $X$  of PTPs is  $O(nm^2)$ .

In the second step “Ideal Polygon Construction”, a non-linear equation set is solved by Newton method, which is quadratic convergent [19]. Assuming that the iterative time is  $k$ , based on the results discussing complexity of Newton method [20], the iterative times depends on the initial points of Newton method and approximation precision. Normally  $k$  is linear relative to  $\log t$  where  $t$  is approximation precision. On the other hand, for each iterative step, several components are calculated with the time complexity listed below:

1. Equation vector  $S_P$  at the iterative point:  $O(n^2)$ .
2. Jacobian matrix at the iterative point:  $O(n^3)$ .
3. Matrix transposition:  $O(n^2)$ .
4. Matrix inversion and matrix multiplication:  $O(n^3)$ .

To sum up, the time complexity of Newton Method is  $O(n^3 \log t)$ , where  $n$  is the total number of line segments and  $t$  is approximation precision. In the third step “Broadcasting Sequence Construction”, the time complexity of Equation

(9) and (11) are  $O(n^3)$  since the calculations only includes matrix transposition, inversion and multiplication. As a result, the time complexity of this method is  $O(nm^2 + n^3k)$  and normally  $k \propto \log t$  where  $t$  is approximation precision.

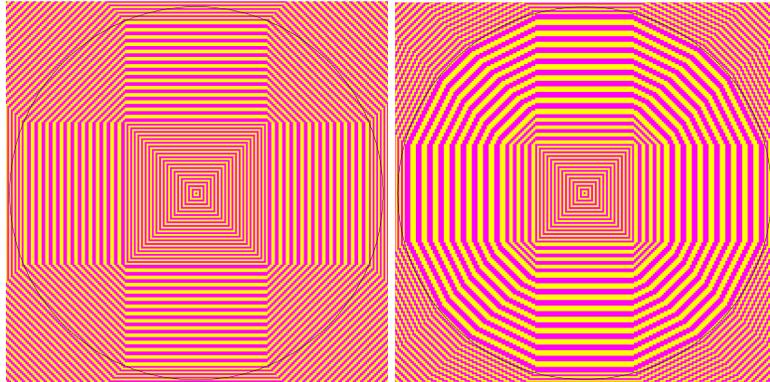
### 5 Construction and comparison

Let us illustrate the results of experiments and a few constructions for specific set of radii. Let us consider the first example for the radius set  $\mathcal{R}_1 = \{r_1 = 2, r_2 = 4\}$  which corresponds to application of Von Neumann neighbourhood (PTP with  $r = 2$ ) and Moore neighbourhood (PTP with  $r = 4$ ). Table 1 shows all intermediate values as well as the optimal broadcasting sequence in this case.

Gradient set	Length vector (X)	Length vector (Y)	Parikh vector (P)	Broadcasting sequence of length 60
$\{0, 1\}$	$\begin{bmatrix} 2 & 0 \\ 0 & 2\sqrt{2} \end{bmatrix}$	$\begin{bmatrix} 0.78 \\ 0.79 \end{bmatrix}$	$\begin{bmatrix} 0.39 \\ 0.28 \end{bmatrix}$	$\{35, 25\}$

**Table 1.** Immediate results of construction process

The image of transmission polygons for the radius set  $\mathcal{R}_1$  is shown on Figure 4 (Left). For this example, the transmission polygons of broadcasting sequence are the same as in case of the neighbourhood sequence. Our broadcasting sequence is optimal and it is matching with optimal error rate of 9.97% in this case.



**Fig. 4.** Constructed polygons for  $\mathcal{R}_1 = \{r_1 = 2, r_2 = 4\}$  (Left) and for  $\mathcal{R}_2 = \{r_1 = 2, r_2 = 4, r_3 = 9\}$  (Right)

The second example uses three types of PTPs for the design of broadcasting sequence. Let the radius set  $\mathcal{R}_2$  be  $\{r_1 = 2, r_2 = 4, r_3 = 9\}$ . Table 2 shows the

intermediate results of each step and Fig. 4 (Right) shows the generated image of transmission polygon. The error rate in this case is 3.03% and the solution is also optimal.

Gradient set	Length vector (X)	Length vector (Y)	Parikh vector (P)	Broadcasting sequence of length 60
$\{0, \frac{1}{2}, 1\}$	$\begin{bmatrix} 2 & 0 & 0 \\ 0 & 0 & \sqrt{5} \\ 0 & 2\sqrt{2} & 0 \end{bmatrix}$	$\begin{bmatrix} 0.52 \\ 0.41 \\ 0.24 \end{bmatrix}$	$\begin{bmatrix} 0.26 \\ 0.08 \\ 0.18 \end{bmatrix}$	$\{29, 10, 21\}$

**Table 2.** Euclidean distance method result

In the Table 3 we provide comparison between the error rate for optimal solution approximating Euclidean circle by broadcasting sequences of length 60 and the sequence generated by our algorithm.

Gradient set	Radius set	Approximated Broadcasting sequence	Error rate of approximated solution	Optimal Broadcasting sequence	Error rate of optimal solution
$0, 1$	$\{2, 4\}$	$\{35, 25\}$	9.97%	$\{35, 25\}$	9.97%
$0, \frac{1}{2}, 1$	$\{2, 4, 9\}$	$\{29, 10, 21\}$	3.03%	$\{29, 10, 21\}$	3.03%
$0, \frac{1}{2}, 1$	$\{5, 16, 18\}$	$\{24, 9, 27\}$	3.03%	$\{24, 9, 27\}$	3.03%
$0, \frac{1}{3}, 1$	$\{2, 25\}$	$\{25, 35\}$	7.55%	$\{0, 60\}$	5.78%
$0, \frac{1}{3}, 1$	$\{5, 25\}$	$\{31, 29\}$	4.51%	$\{31, 29\}$	4.51%
$0, \frac{1}{3}, \frac{1}{2}, 1$	$\{2, 9, 36\}$	$\{36, 9, 15\}$	5.79%	$\{22, 18, 20\}$	2.31%
$0, \frac{1}{3}, \frac{1}{2}, 1$	$\{16, 18, 25\}$	$\{7, 31, 22\}$	3.77%	$\{11, 28, 21\}$	2.98%
$0, \frac{1}{4}, \frac{1}{3}, \frac{1}{2}, 1$	$\{171, 239\}$	$\{34, 26\}$	1.68%	$\{35, 25\}$	1.68%
$0, \frac{1}{4}, \frac{1}{3}, \frac{1}{2}, 1$	$\{233, 239\}$	$\{40, 20\}$	2.15%	$\{11, 49\}$	1.93%

**Table 3.** Experimental results comparing error rates between the optimal and approximate solution

**Conclusion.** The main contribution of the paper is polynomial time algorithm that can find for a fixed set of digital disks the broadcasting sequence which Minkowski sum (i.e. covered area) can approximate Euclidean circles with a small error rate. The efficiency of the method is limited by the ideal polygon (see subsection 4.2) with the same list of gradients as in a given set of digital disks. The error difference between an optimal polygon that can be constructed for a given set of digital disks and a polygon produced by our algorithms can be zero in some cases and also within 3.5% for all tested cases. Our approximation algorithm can calculate the optimal solution (in some cases) or a solution with a small error rate when gradients are repeated for several disks. From the structure of our algorithm follows that the larger set of digital disks we take, the more

common gradients they have leading to smaller error rate between optimal and approximate solutions.

## References

1. M. Worring and A. W. M. Smeulders, "Digitized circular arcs: Characterization and parameter estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 6, pp. 587–598, 1995.
2. B. Nagy and R. Strand, "Approximating euclidean circles by neighbourhood sequences in a hexagonal grid," *Theoretical Computer Science*, vol. 412, no. 15, pp. 1364–1377, 2011.
3. J. Mukherjee, "On approximating euclidean metrics by weighted t-cost distances in arbitrary dimension," *Pattern Recogn. Lett.*, vol. 32, pp. 824–831, Apr. 2011.
4. I. Debledrennesson, S. Tabbone, and L. Wendling, "Fast polygonal approximation of digital curves," *ICPR 2004*, vol. 1, pp. 465–468, 2004.
5. P. Bhowmick and B. B. Bhattacharya, "Approximation of digital circles by regular polygons," in *ICAPR*, pp. 257–267, 2005.
6. P. P. Das, P. P. Chakrabarti, and B. N. Chatterji, "Distance functions in digital geometry," *Information Sciences*, vol. 42, no. 2, pp. 113–136, 1987.
7. P. P. Das and B. N. Chatterji, "Octagonal distances for digital pictures," *Information Sciences*, vol. 50, no. 2, pp. 123–150, 1990.
8. B. Nagy, R. Strand, and N. Normand, "Distance functions based on multiple types of weighted steps combined with neighborhood sequences," *J. Math. Imaging Vis.*, vol. 60, pp. 1209–1219, Oct. 2018.
9. H. Mir-Mohammad-Sadeghi and B. Nagy, "On the chamfer polygons on the triangular grid," *Combinatorial Image Analysis*, pp. 53–65, 2017.
10. J. Farkas, S. Bajak, and B. Nagy, "Approximating the euclidean circle in the square grid using neighbourhood sequences," *Pure Mathematics and Applications*, vol. 17, no. 3-4, pp. 309–322, 2010.
11. T. Nickson and I. Potapov, *Broadcasting Automata and Patterns on  $Z^2$* , vol. 12. Springer, Cham, 2015.
12. R. Martin, T. Nickson, and I. Potapov, *Geometric computations by broadcasting automata*. Kluwer Academic Publishers, 2012.
13. A. Efrima and D. Peleg, *Distributed algorithms for partitioning a swarm of autonomous mobile robots*. Elsevier Science Publishers Ltd., 2009.
14. R. Klette and A. Rosenfeld, "Digital straightness-a review," *Discrete Applied Mathematics*, vol. 139, no. 1, pp. 197 – 230, 2004.
15. D. C. Kozen, *Automata and Computability*. Springer-Verlag New York, Inc., 1997.
16. D. Chen and R. J. Plemmons, "Nonnegativity constraints in numerical analysis," *Symposium on the Birth of Numerical Analysis*, pp. 109–139, 2009.
17. C. L. Lawson and R. J. Hanson, *Solving least squares problems prentice-hall*. SIAM, 1995.
18. L. Shure, "Brief history of nonnegative least squares in matlab." <http://blogs.mathworks.com/loren/2006/>, 2006.
19. O. Ferreira, "Local convergence of newtons method under majorant condition," *Journal of Computational and Applied Mathematics*, vol. 235, no. 5, pp. 1515 – 1522, 2011.
20. P. Batra, "Newton's method and the computational complexity of the fundamental theorem of algebra," *Electronic Notes in Theoretical Computer Science*, vol. 202, pp. 201–218, 2008.