Cellular Lines: An Introduction

Panama Geer, Harry W. McLaughlin and Keith Unsworth

Research Report No :06/2002
       June 2002

ISSN 1174-6696
Applied Computing, Mathematics and Statistics

The Applied Computing, Mathematics and Statistics Group (ACMS) comprises staff of the Applied Management and Computing Division at Lincoln University whose research and teaching interests are in computing and quantitative disciplines. Previously this group was the academic section of the Centre for Computing and Biometrics at Lincoln University.

The group teaches subjects leading to a Bachelor of Applied Computing degree and a computing major in the Bachelor of Commerce and Management. In addition, it contributes computing, statistics and mathematics subjects to a wide range of other Lincoln University degrees. In particular students can take a computing and mathematics major in the BSc.

The ACMS group is strongly involved in postgraduate teaching leading to honours, masters and PhD degrees. Research interests are in modelling and simulation, applied statistics, end user computing, computer assisted learning, aspects of computer networking, geometric modelling and visualisation.

Research Reports

Every paper appearing in this series has undergone editorial review within the ACMS group. The editorial panel is selected by an editor who is appointed by the Chair of the Applied Management and Computing Division Research Committee.

The views expressed in this paper are not necessarily the same as those held by members of the editorial panel. The accuracy of the information presented in this paper is the sole responsibility of the authors.

This series is a continuation of the series “Centre for Computing and Biometrics Research Report” ISSN 1173-8405.

Copyright

Copyright remains with the authors. Unless otherwise stated permission to copy for research or teaching purposes is granted on the condition that the authors and the series are given due acknowledgement. Reproduction in any form for purposes other than research or teaching is forbidden unless prior written permission has been obtained from the authors.

Correspondence

This paper represents work to date and may not necessarily form the basis for the authors’ final conclusions relating to this topic. It is likely, however, that the paper will appear in some form in a journal or in conference proceedings in the near future. The authors would be pleased to receive correspondence in connection with any of the issues raised in this paper. Please contact the authors either by email or by writing to the address below.

Any correspondence concerning the series should be sent to:

   The Editor
   Applied Computing, Mathematics and Statistics Group
   Applied Management and Computing Division
   PO Box 84
   Lincoln University
   Canterbury
   NEW ZEALAND

   Email: computing@lincoln.ac.nz
Cellular Lines: An Introduction

Panama Geer\textsuperscript{a,1,*}, Harry W. McLaughlin\textsuperscript{b}, Keith Unsworth\textsuperscript{c}

\textsuperscript{a}Bryn Mawr College, Department of Mathematics and Computer Science, Bryn Mawr, PA, 19010, USA
\textsuperscript{b}Rensselaer Polytechnic Institute, Department of Mathematical Sciences, Troy, New York 12180, USA
\textsuperscript{c}Lincoln University, Applied Management and Computing Division, P.O. Box 84, Canterbury, NZ

Abstract

This paper provides a definition of a cellular line in a cellular array. It also presents a way of determining whether or not a cell set is a cellular line. Brief statements about existence, uniqueness, and properties of cellular lines are included.

Key words: cellular line definition, cellular array, cellular line string representation, digital line definition, digital geometry

1 Preface

In the engineering community it is common practice to first construct a mathematical model of a physical object and then render the model on a computer display. The rendering process usually involves projecting the object's model onto a plane that has been overlaid with a cellular array, then algorithmically determining which cells to turn on. The engineer understands that the image aids in perception, but the details of the image are not essential for analysis. Supporting the image, there is an "exact" mathematical model available.

While viewing a model of a curve in $\mathbb{R}^2$ on a computer display, an interesting question arises. Are the selected cells in the cellular array limited to modeling

* Corresponding author.

Email addresses: geerp@rpi.edu (Panama Geer), mclauh@rpi.edu (Harry W. McLaughlin), unsworth@lincoln.ac.nz (Keith Unsworth).

1 Partially supported by the National Science Foundation's VIGRE Program (grant DMS-9836466).

Preprint submitted to Elsevier Science 7 June 2002
the underlying curve in $\mathbb{R}^2$, or do they model an equally basic notion: that of a curve in a cellular array? More specifically, is it possible to define the notion of a curve in a cellular array without recourse to the "exact" mathematical model of a curve in $\mathbb{R}^2$?

For example do there exist definitions of lines, circles, parabolas, etc. in a cellular array that are

- independent of the continuum, and
- compatible with engineering practices?

As a starting point in this study, this work provides a definition of a cellular line. Considerable attention has been given to ways of selecting pixels on a computer display, so that they appear straight to the human eye. However, there does not seem to be an agreed upon definition of a cellular line in a cellular array. A definition is offered here.

We suggest that continuum-free geometry in a cellular array may emerge as another approach to geometric modeling.

The need to define a cellular line has arisen in recent studies in which we attempt to use virtual ant algorithms to construct straight lines. This approach is modeled on the work of Dorigo et al. [6] in which ant algorithms are used to find solutions for the traveling salesman problem. In our work, the virtual ants move in a cellular array and they are provided with behavioral rules that permit shortest paths to emerge. Initial experiments with reasonable metrics have produced optimal paths that do not appear straight when viewed on a computer display. The investigations of this paper are motivated by a need to determine when a virtual ant has traversed a "straight" line.

Fig. 1. This cell set depicts the cellular line represented by the string (1 2 1 1 2 1). String representations are discussed later.
2 Introduction

Our goals are

(1) to define the notion of a cellular line (see Figure 1), and
(2) to state an algorithm for determining whether or not a given cell set is a cellular line. ²

The construction of cellular lines is discussed formally in [10]. In Example 1 (below) a cellular line is selected from a pool of candidates. The construction process in [10] is based on this selection process.

There has been much work in the area of line drawing algorithms since the 1960's.³ Recent work includes [2] and [18]. The work can generally be categorized in terms of two distinct approaches: conditional algorithms and structural algorithms. A survey of these approaches is presented in [5]. Structural algorithms are those based on structural properties of string representations of lines. Typically, the line cannot be displayed until the algorithm has finished running. Conditional algorithms take two points as input. Beginning at one of these lattice points, additional lattice points are found according to whether a specified “condition” is satisfied (this condition is usually dependent on a Euclidean line model). In this case, the line can be displayed as it is being generated.

While different from the existing structural approaches, the cellular lines presented here are based on properties of their string representations. These cellular lines are not explicitly generated by an algorithm found in the existing literature.

In 1961 H. Freeman, [7], introduced a method of encoding a geometric configuration in a lattice called “Freeman chain coding.” He stated three properties of chain codes that represent lines. These properties were later formalized using A. Rosenfeld’s “chord property,” [15]. One of the early structural algorithms was presented by R. Brons in [4].

One of the early conditional algorithms was presented by J. Bresenham, [3]. This is often thought of as a benchmark against which other line generation algorithms are compared.⁴ The term “cellular” was used by C. Kim in reference to a line algorithm. He used the phrase “cellular straight line segment”

² It turns out that an algorithm for determining whether or not a given cell set is a cellular line is implicit in the definition.
³ A substantial list of references can be found in [13].
⁴ See [10] for a comparison of the cellular lines defined here and Bresenham’s algorithm.
in describing a new scheme for digitizing curves, [12]. His cellular straight line segments are distinct from the cellular lines presented here.

3 Setting the Stage

We work with two-dimensional rectangular cellular arrays. In defining a cellular line we limit our attention to certain cell sets, called candidate cell sets. Candidate cell sets are connected, column-connected, as short as possible, and they satisfy the second octant convention. Candidate cell sets are not necessary for the definition of a cellular line given below, however, they are introduced to help motivate this definition.

**Connected.** This is modeled by requiring that: except for two cells, all cells in the cell set have exactly two neighbors in the cell set; the two excepted cells, called the initial and terminal cells, each have exactly one neighbor in the cell set.\(^5\) In a cellular array, each cell has eight neighbors.\(^6\)

**Column-connected.** A cell set is column-connected if whenever \(P, Q\) and \(R\) denote cells in the same column of a cellular array and \(Q\) is between \(P\) and \(R\), then \(Q\) belongs to the cell set whenever both \(P\) and \(R\) do.\(^7\) The cell set shown in Figure 3 is not column-connected.

**As short as possible.** This is modeled by requiring that a candidate cell set has a minimum number of cells among all cell sets that have the same

---

\(5\) This is analogous to the definition of a “digital arc” in [15], however here we use cells instead of lattice points.

\(6\) For discussion on other neighborhoods, see for example, [14].

\(7\) A similar statement can be made for row-connected, however we do not explicitly use that idea in this work. The definition of column-connected is related to the definition of the “colinearity property” in [16].
initial and terminal cells and are connected and column-connected. The cell set shown in Figure 4 is not as short as possible.

Fig. 4. This cell set is not a candidate cell set because it is not as short as possible.

Fig. 5. The second octant convention permits terminal cells in the “octant” containing the shaded cells.

**Second octant convention.** This constrains the placement of the initial and terminal cells. We assume that a rectangular coordinate system is imposed on the cellular array so that the chosen initial cell lies in the first row and first column of the cellular array and that the chosen terminal cell lies in the \( c^{th} \) column and the \( r^{th} \) row where \( 1 \leq c \leq r \), see Figure 5.\(^8\) When a cell set satisfies the second octant convention and is as short as possible then there are exactly \( r \) cells in the cell set.

In summary, *candidate cell sets* are those cell sets constrained by the four properties above. Both of the cell sets shown in Figure 1 and Figure 2 are candidate cell sets.

Cellular lines, as they are defined below, are special candidate cell sets. In what follows we justify the choice of the candidate cell set depicted in Figure 1 as a cellular line, while not admitting the candidate cell set depicted in Figure 2.

### 4 Modeling Tool: Strings

In lieu of vector-valued functions that are frequently used as modeling tools to define curves in the Euclidean plane, we use strings (sequences of positive integers) as modeling tools to define cellular lines in cellular arrays. An example is given by the string

\[(1 \ 2 \ 1 \ 1 \ 2 \ 1),\]

\(^8\) The second octant convention is used without loss of generality because it is always possible to impose such a coordinate system after arbitrarily choosing initial and terminal cells in the cellular array.
which records, reading left-to-right, the number of cells in each of the columns of the cell set shown in Figure 1. Thus the first column has 1 cell; the second column has 2 cells; the third column has 1 cell, etc.  

Using the convention illustrated in the above example, it can be seen that the candidate cell sets are exactly the cell sets that have string representations. A cellular line is defined below as a cell set having a string representation with particular properties.

In what follows we work with strings and provide a characterization of string representations of cell sets that we later define to be cellular lines. Preliminary definitions are needed to formalize the notion of a cellular line. To motivate these definitions, we discuss (through an example) the selection of a cellular line with a given terminal cell.

5 Example 1

For the cell set in Figure 1 the terminal cell is in the 6th column and the 8th row i.e., c = 6 and r = 8. The length of the corresponding string is 6, i.e., it has six components. The sum of the components of the string is 8. There are 21 such strings and hence 21 corresponding cell sets satisfying the four conditions in §3. Exactly one of them, shown in Figure 1, is selected as a cellular line.

Three of the 21 strings are listed below.

\[(1 2 1 1 2 1) \quad (2 1 1 1 2 1) \quad (1 1 3 1 1 1)\]

They represent the cell sets shown in Figure 1, Figure 2 and Figure 6 respectively. There are two primary distinctions between the cell sets depicted in these figures.

---

9 In the string \((1 2 1 1 2 1)\), we represent two cells by the number 2. The concept of representing multiple cells by one symbol is analogous to the notion of a “span” in [1] or a “run” in [17].

10 A mathematically precise definition of string representations is found in [11].

11 This is analogous to defining a Euclidean straight line in terms of a function that has the property of linearity.

12 If the string contains six components that sum to eight then the string must have either two 2's and four 1's or one 3 and five 1's. There are 21 such strings, i.e. \(21 = \binom{5}{2} + \binom{8}{2}\). If each of the components of the string were the symbol 1, their sum would be six; changing two of the symbols to 2 or one of the symbols to 3 results in a sum of eight.
The third column in Figure 6 is significantly longer than the other columns in that figure; whereas the column lengths in Figure 1 and Figure 2 are nearly equal. 13

Even though the column lengths in Figure 2 are nearly equal, the longer columns in Figure 2 are not as evenly distributed among the shorter columns as they are in Figure 1. 14

Fig. 6. This cell set corresponds to the string (1 1 3 1 1 1).

Our definition of a cellular line reflects the above two observations; additionally it reflects notions of even distributions in higher order constructs. 15 The human eye can readily determine when column lengths vary substantially, as in Figure 6. The human eye can also frequently determine when columns of different lengths are not evenly distributed, as in Figure 2. However, the human eye cannot generally detect structural properties represented by the higher order string constructs developed below. This is analogous to higher order derivatives that characterize structural properties of planar curves, properties not detectable with the human eye. Just as some work is needed to develop the notions of derivatives and derivatives of derivatives in order to define special curves in the plane, we invest some effort in developing the notion of derived strings of different orders to define cellular lines.

Construction of Candidate Strings. Given $c = 6$ and $r = 8$, the candidate strings are the strings: (i) over the alphabet of positive integers, (ii) of length six, (iii) whose components sum to eight. There are 21 candidate strings. See Footnote 12. The candidate cell sets corresponding to three of the candidate strings are shown in Figures 1, 2, and 6.

Construction of Zero Order Derived Strings. Requiring that the column lengths of the candidate cell sets be nearly equal, means that two components

---

13 In both Figure 1 and Figure 2 column lengths vary by no more than one cell.
14 In [8], Freeman stated three properties for strings that represent straight lines. Requiring columns to be “nearly equal” and “evenly distributed” satisfies Freeman’s properties.
15 The structural algorithm presented by Brons, in [4], uses an analogous hierarchical approach to generate a string representation of a straight line on a grid with Freeman coding. However, our definition of a cellular line does not arbitrarily place “the least occurring symbol” at “the end of the period” as Brons’ algorithm dictates.
of each of the corresponding strings must be 2 and four of the components must be 1, resulting in 15 such strings, i.e. $15 = \binom{6}{2}$.

The zero order derived strings are those candidate strings over the alphabet \{1, 2\}. Three of the 15 zero order derived strings are listed below.

$$(121121) \quad (112121) \quad (221111)$$

For the zero order derived strings, the symbol 2 is called the minority symbol and the symbol 1 is called the majority symbol because the symbol 2 appears less frequently than the symbol 1.

The zero order derived strings listed above represent the cell sets shown in Figure 1, Figure 7 and Figure 8 respectively.\(^{16}\) Notice that the alphabet underlying the zero order derived strings, consisting of two consecutive positive integers, models the desired property that column lengths be nearly equal.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{fig7.png}
\caption{Fig. 7. This cell set corresponds to the zero order derived string $(112121)$.}
\end{figure}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{fig8.png}
\caption{Fig. 8. This cell set corresponds to the zero order derived string $(221111)$.}
\end{figure}

**Construction of First Order Derived Strings.** These are generated by requiring that columns of different lengths be evenly distributed. This leads to the following considerations. Among the 15 zero order derived strings there are some that represent candidate cell sets with columns of length 2, evenly distributed among the columns of length 1. That is, there are zero order derived strings having three substrings of 1's (majority symbols) partitioned by the two 2's (minority symbols). There are three such strings, i.e. $3 = \binom{3}{2}$.

$$(121121) \quad (112121) \quad (121211)$$

For these strings, we say that the 2's *fully partition* the strings.\(^{17}\)

\(^{16}\) The string representation of the cell set in Figure 2 is also a zero order derived string.
\(^{17}\) A string is fully partitioned when the number of substrings of majority symbols is maximal.
Each first order derived string is formed from a zero order derived string that is fully partitioned by the 2's (the minority symbol). A fully partitioned string is transformed into a first order derived string in two steps.

i. Each substring of 1's (majority symbol) is replaced by the integer that counts the number of 1's in the substring.

ii. The minority symbols (the original 2's) are deleted.

For example, \((1 1 2 1 2 1) \rightarrow (2 1 1 2 1) \rightarrow (2 1 1)\), where 2 denotes a 2 appearing in the zero order derived string. The three first order derived strings are shown below along with their predecessor zero order derived strings.

<table>
<thead>
<tr>
<th>First Order Derived String</th>
<th>Zero Order Derived String</th>
</tr>
</thead>
<tbody>
<tr>
<td>((1 2 1))</td>
<td>((1 2 1 2 1))</td>
</tr>
<tr>
<td>((2 1 1))</td>
<td>((1 1 2 1 2 1))</td>
</tr>
<tr>
<td>((1 1 2))</td>
<td>((1 2 1 2 1))</td>
</tr>
</tbody>
</table>

The first order derived strings reflect an even distribution of columns of different heights in candidate cell sets. For example, the first order derived string, \((2 1 1)\) or \((\text{two, one, one})\), reflects: the first two columns of length 1 (denoted by the 1 1 at the beginning of the zero order derived string) are separated from one column of length 1, which in turn is separated from one column of length 1.

Construction of Second Order Derived Strings. Geometric motivation for the construction of second order derived strings is delayed. Each first order derived string contains two 1's and one 2. Following the same procedure as above, the 2's partition each first order derived string into substrings of 1's. The notion of even distribution of the 2's among the 1's leads to the understanding that the 2's partition some of the first order derived strings into two substrings of 1's; in this case, on the string \((1 2 1)\). This first order derived string is said to be fully partitioned by the 2.

Each second order derived string is formed from a first order derived string that is fully partitioned by the 2 (the minority symbol). The transformation from a first order derived string to a second order derived string occurs in two steps.

i. Each substring of majority symbols (1's) is replaced by the integer that counts the number of majority symbols (1's) in the substring.

ii. The minority symbols (original 2's) are deleted.

There is 1 second order derived string, i.e. \(1 = \binom{2}{2}\). It is shown below with its predecessor first and zero order derived strings.
Construction of Third and Higher Order Derived Strings. Using the above process, there is exactly one third order derived string. \(^{18}\) It is shown below with its predecessor second, first and zero order derived strings.

\[
\begin{array}{c|c|c|c}
\text{Third Order} & \text{Second Order} & \text{First Order} & \text{Zero Order} \\
(2) & (1 1) & (1 2 1) & (1 2 1 2 1)
\end{array}
\]

Fourth and higher order derived strings will all exist and be of length one. \(^{19}\) Generating them will not identify other zero order derived strings.

Thus the string

\[(1 2 1 1 2 1)\]

is \textit{defined} to be the string representation of the cellular line determined by the terminal cell in the 6\(^{th}\) column and the 8\(^{th}\) row (Figure 1).

\[\text{Fig. 9. The partitioning columns are shaded.}\]
\[\text{Fig. 10. The partitioning columns are deleted.}\]
\[\text{Fig. 11. The partitioning columns are collapsed and shaded.}\]
\[\text{Fig. 12. The partitioning column is deleted and collapsed.}\]

Geometric Motivation for Second Order Derived Strings. Using a delete-collapse process, the second order derived string is obtained from the first order derived string. Similarly the first order derived string is obtained from the zero order derived string.

Figure 9 shows the cell set (cellular line) represented by the zero order derived string, \((1 2 1 1 2 1)\), with the cells in the two partitioning columns shaded. Deleting the cells in the partitioning columns (Figure 10) and "collapsing" the resulting cell set yields the cell set of Figure 11. It represents the first order

\(^{18}\) Notice that there is no minority symbol in the second order derived string, so none is deleted. We simply replace the two 1's with the integer 2.

\(^{19}\) This is analogous to the fact that the \((n + 1)^{st}\) and higher order derivatives of an \(n^{th}\) degree polynomial are zero.
derived string, \((1 2 1)\). In Figure 11 the shaded cells lie in the one partitioning column corresponding to the first order derived string. Deleting the cells in this partitioning column and "collapsing" the resulting cell set yields the cell set of Figure 12. It represents the second order derived string, \((1 1)\). In this case there are no partitioning columns to shade.

6 Definitions

The steps used in Example 1 can be described as a search among 21 candidate strings for the one string that has all four \((0, 1^{st}, 2^{nd}, 3^{rd})\) order derived strings. In order to identify different order derived strings, the notion of fully partitioning a string by one of its elements was used. In this section, we formalize the two notions of fully partitioned and derived strings. Using these definitions we offer a definition of a cellular line in terms of different orders of derived strings and conclude with several examples.

**Fully Partitioned Strings.** In a string over a two-symbol alphabet one of the symbols occurs not more frequently than the other. It is designated as the string's *minority symbol*. The other symbol is designated as the string's *majority symbol*.²⁰

A string of integers is said to be *fully partitioned* when the following conditions hold.

i. It is a finite string over an alphabet of two consecutive positive integers.

ii. When the two symbols do not occur the same number of times, the minority symbol partitions the string into \(|m| + 1\) non-empty substrings of majority symbols, called the *majority symbol substrings*, where \(|m|\) denotes the number of times the minority symbol occurs in the string.

iii. When the two symbols occur the same number of times, the minority symbol partitions the string into \(|m|\) non-empty substrings of majority symbols, also called majority symbol substrings.²¹, ²²

---

²⁰ When the two symbols occur the same number of times, either can be designated as the minority symbol.
²¹ In this case each symbol alternates and the substrings are of length 1. It is also understood that when both symbols appear the same number of times, either one of them can be selected as the minority symbol.
²² The string \((4 3 4 4 3 4)\) is fully partitioned. Its minority symbol is 3, which occurs twice. The symbol 3 partitions the string into three \((|m| + 1)\) substrings of majority symbols, 4's. The string \((4 3 4 3 4 3)\) is fully partitioned. Either of the two symbols, 3 or 4 can be designated as the minority symbol. In each case, the minority symbol partitions the string into three \(|m|\) substrings of majority symbols.
It is understood that if the string contains only one of the two symbols from the alphabet, the missing one is designated the minority symbol. In this case the minority symbol partitions the string into one substring, namely the whole string. Thus any finite string with only one distinct symbol is fully partitioned.

**j**th Order Derived Strings.

Let $S$ denote a string over some alphabet. If $S$ is a string over an alphabet consisting of two consecutive positive integers, denote its zero order derived string by $S^{(0)}$ and define it by $S^{(0)} = S$.

For all integers, $j \geq 1$, if $S^{(j-1)}$ is fully partitioned,

i. replace each of the majority symbol substrings of $S^{(j-1)}$, by its length,
ii. delete the minority symbols of $S^{(j-1)}$,
iii. define the resulting string to be the $j$th order derived string of $S$ and denote it by $S^{(j)}$.

If for some $n$, $\text{length}(S^{(n)}) = 1$, then for all $k > n$, $S^{(k)}$ exists and $S^{(k)} = (1)$. Further, if all derived strings of $S$ exist, then for some $n$, $\text{length}(S^{(n)}) = 1$. This follows from the observation that if $\text{length}(S^{(j-1)}) > 1$ and if $S^{(j)}$ exists, then $\text{length}(S^{(j-1)}) > \text{length}(S^{(j)})$.

If $S$ is not a string over an alphabet consisting of two consecutive positive integers, then $S^{(0)}$ is not defined. Further, if $S^{(j-1)}$ is not fully partitioned (for some $j \geq 1$), then $S^{(j)}$ and higher order derived strings are not defined.

**Cellular Line.**

A cellular line is a cell set that is represented by a string, $S$, such that the $j$th order derived string of $S$ exists for all $j \geq 0$.\(^{23}\)

The definition implicitly defines an algorithm for determining whether or not a given cell set is a cellular line. If a cell set can be represented by a string, $S$, then for all $j \geq 0$ determine whether or not $S^{(j)}$ exists. Notice that this is a finite process since there is a $j$ such that either $S^{(j)}$ does not exist or $\text{length}(S^{(j)}) = 1$.

In particular we note that a cellular line may be defined in terms of strings only, omitting the geometry of cell sets in cellular arrays.\(^{24}\)

**Examples.** The examples below show how the definition of a cellular line implicitly defines an algorithm for determining whether a given cell set is a cellular line.

\(^{23}\) There exists an integer, $n$, such that for $n \leq k$, $S^{(k)}$ exists and $\text{length}(S^{(k)}) = 1$.

\(^{24}\) This is analogous to the classical definition of a curve in a topological space; a curve is a continuous function from the unit interval into the topological space.
Example 2.

![Figure 13](image)

This cell set corresponds to string (1211212121121121).

$S = (1211212121121121)$.

$S^{(0)} = (1211212121121121)$.

$S^{(0)}$ is fully partitioned.

$S^{(1)} = (12112121121)$.

$S^{(1)}$ is fully partitioned.

$S^{(2)} = (121)$.

$S^{(2)}$ is fully partitioned.

$S^{(3)} = (11)$.

$S^{(3)}$ is fully partitioned.

$S^{(4)} = (2)$.

length($S^{(4)}$) = 1.

The cell set is a cellular line.

Example 3.

![Figure 14](image)

This cell set corresponds to string (1211212121121121).

$S = (1211212121121121)$.

$S^{(0)} = (1211212121121121)$.

$S^{(0)}$ is fully partitioned.

$S^{(1)} = (1212111)$.

$S^{(1)}$ is fully partitioned.

$S^{(2)} = (112)$.

$S^{(2)}$ is not fully partitioned.

The cell set is not a cellular line.

Example 4.

![Figure 15](image)

This cell set corresponds to string (3 4 3 4).

$S = (3 4 3 4)$.

$S^{(0)} = (3 4 3 4)$.

$S^{(0)}$ is fully partitioned.

$S^{(1)} = (11)$.

$S^{(1)}$ is fully partitioned.

$S^{(2)} = (2)$.

length($S^{(2)}$) = 1.

The cell set is a cellular line.
Example 5.

\[
\begin{align*}
S &= (8), \\
S^{(0)} &= (8), \\
\text{length}(S^{(0)}) &= 1.
\end{align*}
\]

The cell set is a cellular line.

Fig. 16. This cell set corresponds to string (8).

7 Existence, Uniqueness, and Properties

**Existence.** It is clear that, given a string representation of a cellular line, one can determine the terminal cell of the cellular line. Less clear is the fact that there exists a cellular line with an arbitrarily selected terminal cell. The verification of this assertion appears in [11]. There it is shown that:

For integers \( c \) and \( r \), \( 1 \leq c \leq r \), there exists a cellular line with an initial cell in the first column and first row and a terminal cell in the \( c^{th} \) column and \( r^{th} \) row. It is represented by a string:

i. over the alphabet \( \{\lfloor \frac{r}{c} \rfloor, \lfloor \frac{c}{c} \rfloor + 1\} \),
ii. whose length is \( c \)
iii. whose components sum to \( r \).

For example, when \( c = 13 \) and \( r = 31 \), the string representation of the corresponding cellular line is a string of length 13 over the alphabet \( \{2, 3\} \). Using a construction, similar to that of Example 1, one finds the string representation to be \( (2 3 2 2 3 2 3 2 3 2 3 2 3 2) \).

**Uniqueness.** The above definition leaves open the possibility that there exists more than one cellular line with a given terminal cell. The strings \( (1 2 1 2) \) and \( (2 1 2 1) \) both represent cellular lines determined by the cell in the \( 4^{th} \) column and \( 5^{th} \) row.

It is shown in [11] that there exist at most two cellular lines with the same initial and terminal cells.
Palindromes. It follows from the Uniqueness Theorem in [11], that when there is only one cellular line determined by a terminal cell, the string representation is a palindrome, [10], [11]. Further, when there are two cellular lines, determined by a terminal cell, the string representations are not palindromes. The two strings, determined by fixed values of $c$ and $r$, are dual to each other. For example the cell set shown in Figure 17 can be represented by both the strings $(1\ 2\ 1\ 2)$ and $(2\ 1\ 2\ 1)$ depending upon the choice of coordinate system.

![Figure 17](image17.png)

Fig. 17. This cell set depicts the cellular line denoted by the string $(1\ 2\ 1\ 2)$ or $(2\ 1\ 2\ 1)$ depending on the choice of coordinate system.

Finally it is observed that the string $(1\ 2\ 2\ 1)$ is a palindrome but does not represent the cellular line determined by $c = 4$ and $r = 6$.

Comparisons. The following is a list of comparisons between cellular lines in a cellular array and chords in $R^2$.

- Chords (straight line segments) in $R^2$ have the property that their connected subsets are also straight lines. The chord determined by two points of a chord in $R^2$ is a subset of the original chord. This is not always the case for cellular lines. For example, the cellular line shown in Figure 18 is represented by the string, $(1\ 2\ 1\ 2\ 1)$. It includes cells $A$ and $B$ but not cell $C$. However, the cellular line with initial cell $A$ and terminal cell $B$, represented by the string $(1\ 2\ 1)$, does include cell $C$.

![Figure 18](image18.png)

Fig. 18. This cell set is denoted by the string $(1\ 2\ 1\ 2\ 1)$.

- Chords in $R^2$ have the property that they belong to exactly one (infinite) straight line. Cellular lines do not have such a property. Every cellular line
has an infinite number of extensions that are cellular lines. This is discussed in [10].

- Between every two points in \( \mathbb{R}^2 \) there exists a unique chord in \( \mathbb{R}^2 \). Cellular lines do not have this property. For example, the strings (1 2 1 2) and (2 1 2 1) represent two cellular lines with their terminal cell in the 4th column and the 6th row.
- A chord in \( \mathbb{R}^2 \) is the unique geodesic between its two end points. There does not appear to be a direct cellular line analogue.

8 Ongoing Studies and Conclusions

The following list shows some ongoing studies that have emerged from this study of cellular lines.

- The questions of existence, uniqueness, and construction of cellular lines have been put on secure mathematical foundations. They appear in [11] and [10].
- A notion of curvature of strings, used to characterize cellular lines, has been formulated. It is discussed in [10].
- One can "integrate" string representations of cellular lines, in two dimensional cellular arrays, in order to define higher order cellular curves (quadratic, cubic, etc.) This has been done and is reported in [10].
- There is not a unique method of extending a cellular line. This is discussed in [10]. Ways of defining infinite cellular lines are under investigation.
- There are symmetries within the set of all cellular lines. Hexadecant symmetry is mentioned by Boyer et al., [1]. These are discussed in [10].
- The construction of cellular lines reminds one of a growing process. This idea leads to the possibility of being able to grow "arbitrary" cellular curves. An algorithm that models the growing process has been formulated but substantial experimentation is needed.
- An "ant algorithm" for finding cellular lines has been formulated and tested. It is described in [9] and will appear elsewhere.
- Some progress has been made toward defining a cellular line in a three-dimensional cellular array. This is briefly mentioned in [10].

In the preface, we introduce the idea of defining geometric objects in a cellular array such that these definitions are independent of the continuum while being compatible with engineering practices. The work presented here provides a starting point: a definition of a cellular line without the aid of Euclidean constructs. It also provides a way of determining whether or not a cell set is a cellular line.
Acknowledgements

The authors would like to thank Ken Brodlie for his helpful comments during the writing of this paper.

References


