Identification of Hybrid Cellular Automata Using Image Segmentation Methods

Y. ZHAO\textsuperscript{1}, H.M. GUO\textsuperscript{2} AND S.A. BILLINGS\textsuperscript{1}

\textsuperscript{1}Department of Automatic Control and System Engineering, University of Sheffield, UK. E-mail: s.billings@shef.ac.uk
\textsuperscript{2}School of Mechanical, Aerospace and Civil Engineering, University of Manchester, UK.

Received: March 5, 2011. Accepted: November 23, 2011.

When given a complex cellular automata (CA) system, especially a real system, the transition rule over the whole evolution is often not uniform, which means that different spatial positions may have different rules at the same time. Currently, most methods for the identification of CA are only suitable for systems with uniform rules. Therefore, it is necessary to develop an algorithm which can detect the region with a specific rule or partition each region with different rules for a hybrid CA system. Current methods of identification could then be applied to each region to identify the CA model. By mapping the realistic CA pattern to a virtual image, this paper first introduces two popular image segmentation algorithms to aid the identification of hybrid CA. A popular nonlinear filter in image processing, the median filter, is then proposed to remove noise in the segmented image to avoid over-estimation. Two examples, including a one-dimensional and a two-dimensional CA system are then employed to demonstrate the algorithms. It is shown that the results are encouraging by comparing the original rule distribution graph and the detected rule distribution graph, and comparing the reconstructed patterns and the observed patterns.

Keywords: cellular automata, hybrid, image segmentation, identification, region growing, spatio-temporal

1 INTRODUCTION

Most previous studies on the identification of CA have concentrated on uniform cellular automata, or CA where the transition rule is identical at all positions over the whole lattice and at all times during the evolution...
However, when studying an unknown real system, it may be unwise to assume that the rule representing the system holds over the whole image. Hence, recently, the studies of hybrid cellular automata has attracted more and more investigations [6, 7, 3, 10, 12], which try to represent a complex system using multi-rules. However, very few of these methods can solve the inverse problem - that is identifying the hybrid cellular automata rule given the observed data.

It has been shown in [22] that methods for the identification of CA and excitable media can work when given observed data from a uniform rule. However, if for example the observed data are mixed by two rules with fifty percent probability respectively, an incorrect model would obviously be obtained using the current methods.

This paper introduces a new method for the identification of hybrid CA initially which partitions the regions with different rules initially by employing the methods from image processing, and then applies the methods in [22] for the identification of CA with a uniform rule in each separated region to obtain the final hybrid rule or model. The paper is organized as follows. Section 2 maps the observed CA pattern into a virtual image, which forms the object to be processed in the next steps. Two algorithms to detect the rule distribution derived from image segmentation are then introduced in Section 3. Two examples, including a one-dimensional and a two-dimensional CA system are then employed in Section 4 to demonstrate the new algorithms. Conclusions are given in Section 5.

2 RELATIONSHIP BETWEEN HYBRID CELLULAR AUTOMATA AND THE VIRTUAL IMAGE

2.1 Hybrid Cellular automata

A uniform cellular automata is composed of three parts: a neighbourhood, a local transition rule and a discrete lattice structure. The local transition rule updates all cells synchronously by assigning to each cell, at a given step, a value that depends only on the neighbourhood. The definition of hybrid CA in this paper is that the different positions on the lattice might have different transition rules at the same time, but the rules on the same position are identical at different times. As an example a one-dimensional hybrid CA with a periodic boundary is demonstrated in Figure 1. In this example, there are three different transition rules in different regions. As shown in Figure 1.(a), the regions labelled by 2, 4, 6 have a uniform rule respectively without noise. Mixed rules exist in regions labelled by 1, 3, 5, 7. For example, the region labelled by 3 is a mix of Rule A and Rule B to represent the transition from Rule A to Rule B. In this transition regions the probability of Rule A at spatio position $x$ is equal to $\frac{x_2 - x}{x_2 - x_1}$, where $x_1 \leq x \leq x_2$. Figure 1.(b) shows the generated pattern by a hybrid CA on a $300 \times 300$ lattice with three rules - $R60$, $R105$ and $R129$.
2.2 Mapping the Pattern of Hybrid CA into a Virtual Image

An image is composed of pixels on a lattice, and a CA is composed of cells on a lattice, which makes the possibility of applying segmentation methods from image processing to the identification of hybrid CA.

Image segmentation is one of the most well addressed problems in computer vision. The complexity of such a problem varies according to the application, where in the most general case one would like to partition an image into regions with consistent properties. Such properties can be either visual or geometric. An example of image segmentation derived from the application in medical area is shown in Figure 2, where the interested part, the white region in the center, is partitioned by the image segmentation and highlighted in Figure 2(b).

If partial segmentation is the goal, an image is divided into separate regions that are homogeneous with respect to a chosen property such as brightness, colour, reflectivity, texture, etc. Hence, the key to apply the segmentation methods in image processing to rule distribution detection for hybrid CA is selecting an appropriate property which could represent a feature of the rule. However, it may be impossible or very difficult to find the correct rule for every single cell in an observed CA pattern, because when considering a two-dimensional CA for example on a \( n_x \times n_y \) lattice and collecting \( n_t \) time steps of sampled data, only \( n_t \) data can be used to detect the rule for every single cell, but there could be \( 2^{10} = 1024 \) possible cases (the definition of case could be represented by Definition 1 because if the neighbourhood of the CA is the Moore neighbourhood).
Definition 1: Consider a two-dimensional CA for example, when the cell at position \((x, y)\) and at time step \(t\) is denoted as \(c(x; y; t)\), then a case is defined as a pair of \([f(N[c(x; y; t − 1)]), c(x; y; t)]\), where \(N[c(x; y; t − 1)]\) is the neighbourhood of the cell \(c(x; y)\) at time step \(t − 1\) and \(f(N[c(x; y; t − 1)]) = c_1 + 2c_2 + \ldots + 2^{m-1}c_m\) assuming \(N[c(x; y; t − 1)] = \{c_1, c_2, \ldots, c_m\}\). For example, if the state value of the updated cell \(c(x; y; t)\) is 1 and the state of the neighbourhood \(N[c(x; y; t − 1)]\) is \([0, 1, 1]\), the case can be described as \([6, 1]\).

Using a single cell it could be impossible to include all 1024 cases when \(n_t < 1024\). Even if \(n_t\) is slightly larger than 1024, the sampled data will not be enough because some cases might be repeated and some cases may never happen. If a rule is obtained under such a situation, the result could be over-estimated because of loss of data. The solution to solving this problem is that a pixel in a virtual image can be created by mapping a rectangular region with size \(n_b \times n_b\) in the CA pattern, and this will be referred to as a block. Consider the example illustrated by Figure 3, where a CA pattern on a \(n_x \times n_y\) lattice is mapped into a virtual image with \(n_x n_b \times n_y n_b\) pixels (\(n_b = 4\) in this example). Because there are \(n_b \times n_b \times n_t\) data that can be utilized the possibility of obtaining the rule of the considered block should be larger because of the larger possibility of including all cases. However, if \(n_b\) is selected to be too large, then obviously the segmentation would be less accurate and the probability of multi-rules in one block may be large. This in turn could lead to the difficulty in calculating the property of the block. This paper introduces a parameter named \(T_u\), which represents a noise-immune ability. Consider a block which has \(n\) rules denoted by \(r_1, r_2, \ldots, r_n\). If there is a rule \(r_i\) whose percent of occupancy in that block is larger than \(T_u\), this block will be marked as obeying \(r_i\). If there is no such rule, this block should be marked as unidentifiable. Obviously, \(T_u\) should be chosen between 0.5 to
I DENTIFICATION OF HYBRID CA USING IMAGE SEGMENTATION METHODS

FIGURE 3
An illustration for mapping the CA pattern to a virtual image, where gray cells in (a) denote the considered block and the gray cell in (b) represents the mapped pixel ($n_b = 4$ in this example).

1. The smaller $T_u$ is, the more probability there is of identifying the block, but there is more risk of getting a wrong rule. Hence, there is a clear tradeoff between the accuracy of segmentation and the property calculations, and a tradeoff in the block property between the accuracy and the identifiability.

   The crucial step of the mapping procedure is how to calculate the property of each block in a CA pattern. Consider a $n_b \times n_b$ region in the lattice of a two-dimensional hybrid CA and assume $n_t$ time steps of data are sampled. The procedure of property determination can be summarized by:

   1. Detect the neighbourhood of the considered block. Because in many applications little information about the structure of the rule will be known, the neighbourhood of the rule must be determined prior to generating the model. A neighbourhood detection algorithm using mutual information was proposed in [21], which will be employed in this paper to detect the neighbourhood of the current block. If multi-rules exist in this block, the maximal neighbourhood should be chosen.

   2. Collect the cases using the detected neighbourhood and calculate the counter $C_{iy}$ for each case, where $C_{iy}$ denotes the number of occurrences of the case \{f(N_i), y\}.

   3. Extract the significant cases. Because multi-rules may exist or noise may be included, there could be some paradoxical cases, such as \{0, 0\} and \{0, 1\}. To get a uniform rule in the considered block, one of the above two cases must be discarded. Assume there are two paradoxical cases \{f(N_i), 0\} and \{f(N_i), 1\}. If $\frac{C_{i0}}{C_{i0} + C_{i1}} \geq T_u$, the case \{f(N_i), 1\} should be discarded, and if $\frac{C_{i1}}{C_{i0} + C_{i1}} \geq T_u$, the case \{f(N_i), 0\} should be discarded, else this region should be marked as unidentifiable. Normally, $T_u$ is chosen between 0.65 to 0.95.
4. Assume step 3 extracted the cases which are \( \{f(N_i), y_i\} (i = 1, m) \), where \( m \) denotes the number of the cases. The property of each block can then be calculated by

\[
P_v = \sum_{i=1}^{m} y_i \times 2^{f(N_i)}
\]

Finally, the mapping procedure between the pattern of hybrid CA and the virtual image can be summarized as:

Consider a \( n_x \times n_y \) hybrid CA pattern and denote the selected block size as \( n_b \). If the currently considered block is identifiable, the property of the considered pixel in the image is assigned by the value \( P_v \) of the corresponding block. If the block is unidentifiable, the property of the corresponding pixel in the image should be assigned by a special value, such as 0. The size of the mapped virtual image would be \( \frac{n_x}{n_b} \times \frac{n_y}{n_b} \).

3 REGION SEGMENTATION

The main goal of segmentation is to divide an image into parts that have a strong correlation with objects or areas. This section discusses the segmentation techniques that are based on finding the parts directly. Let \( R \) represent the entire image region. The segmentation may be viewed as a process that partitions \( R \) into \( n \) subregions, \( R_1, R_2, \ldots, R_n \), such that

\[
\begin{align*}
(a) \quad & \bigcup_{i=1}^{n} R_i = R, \\
(b) \quad & R_i \text{ is a connected region, } i = 1, 2, \ldots, n, \\
(c) \quad & R_i \cap R_j = \phi \text{ for all } i \neq j, \\
(d) \quad & P(R_i) = \text{TRUE for } i = 1, 2, \ldots, n, \\
(e) \quad & P(R_i \cup R_j) = \text{FALSE for } i \leq j
\end{align*}
\]

where \( P(R_i) \) is a logical predicate defined over the points in set \( R_i \), and \( \phi \) is the null set. Condition (a) indicates that the segmentation must be complete; that is, every pixel must be in a region. The second condition requires that points in a region must be connected. Condition (c) indicates that the regions must be disjoint. Condition (d) deals with the properties that must be satisfied by the pixels in a segmentation region. In the current application, \( P(R_i) = \text{TRUE} \) implies the pixels in the region \( R_i \) have the same properties. Finally, condition (e) indicates that regions \( R_i \) and \( R_j \) are different in the sense of predicate \( P \). Based on conditions expressed by Equation (2), two popular image segmentation methods are introduced in the following sections.
3.1 Region Growing

Region growing is a well known technique for image segmentation [16, 17, 18]. It postulates that neighbouring pixels within the same region have similar intensity values. The general idea of region growing is to group pixels with the same or similar intensities to one region according to a given homogeneity criterion. More precisely, region growing starts with a set of pre-specified seed pixel(s) and grows from these seeds by merging neighbouring pixels whose properties are most similar to the pre-merged region. Typically, the homogeneity criterion is defined as the difference between the intensity of the candidate pixel and the average intensity of the pre-merged region. If the homogeneity criterion is satisfied, the candidate pixel will be merged to the pre-merged region. The procedure is iterative: at each step, a pixel is merged according to the homogeneity criterion. This process is repeated until no more pixels are assigned to the region.

Consider the mapped virtual image from a CA pattern. Region growing in segmentation of hybrid CA can be summarized by:

1. Start by choosing an arbitrary seed pixel and compare it with its neighbouring pixels, which can be illustrated by Figure 4.(a), where the black cell denotes the seed and the gray cells denote the seed’s neighbouring pixels. In this example, the neighbourhood structure is von Neumann.

2. The region is grown from the seed pixel by adding in neighbouring pixels that have the same properties, increasing the size of the region, seen in Figure 4.(b).

3. When the growth of one region stops we simply choose another seed pixel which does not yet belong to any region and start again, for example in Figure 4.(c). Figure 4.(d) shows the result when two regions have been partitioned.

4. This whole process continues until all pixels belong to some region.

Region growing is an effective but time consuming method. What this paper proposes is the principle of this method. Many modified region growing methods have been presented recently to improve the efficiency by adjusting the search strategy [8]. If only one region is to be segmented, this method can be simplified by choosing only one seed in the interested region. The procedure would be stopped once the boundary of this region has been detected.

3.2 The Region Splitting and Merging Method

The procedure discussed in the previous section grows regions starting from a given set of seed pixels. An alternative is initially to subdivide an image into a set of arbitrary, disjoined regions and then merge and/or split the regions in an attempt to satisfy the conditions stated in Equation (2). A region splitting and merging algorithm that iteratively works toward satisfying these constraints is explained as follows.
FIGURE 4
A simple example of the region growing method. (a) Starting from a chosen seed pixel in the considered region; (b) The region is grown from the seed pixel by adding in neighbouring pixels that have the same properties; (c) The result for the first region after segmentation and the start of the second region; (d) The results for the two regions after segmentation.

Let $R$ represent the entire image region and select a predicate $P$ as discussed in Equation (2). For a hybrid CA, $P(R_i) = \text{TRUE}$ means all the cells in $R_i$ have the same CA transition rule. Assume a square image, one approach for segmentation of $R$ is to subdivide it successively into smaller and smaller quadrant regions such that, for any region $R_i$, $P(R_i) = \text{TRUE}$. That is, if $P(R_i) = \text{FALSE}$, the image is divided into quadrants. If $P$ is FALSE for any quadrant, that quadrant is subdivided into sub-quadrants, and so on. This particular splitting technique has a convenient representation in the form of a so-called quadtrees, which is demonstrated by Figure 5.(a). Figure 5.(b) shows the geographical position of each symbol in the quadtrees. Note that the root of the tree corresponds to the entire image and each node corresponds to a subdivision. In this case, only $R_4$ was subdivided further.

If only splitting is used, it is likely that the final partition would contain adjacent regions with identical properties. This may be remedied by allowing merging, as well as splitting. In order to satisfy the constraints in Equation (2),
adjacent regions whose combined pixels satisfy the predicate $P$ are merged; that is, two adjacent region $R_i$ and $R_j$ are merged only if $P(R_i \cup R_j) = \text{TRUE}$.

The preceding discussion may be summarized by the following procedure in which, at any step:

1. Split into four disjointed quadrants any region $R_i$ where $P(R_i) = \text{FALSE}$.
2. Merge any adjacent regions $R_j$ and $R_k$ for which $P(R_j \cup R_k) = \text{TRUE}$.
3. Stop when no further merging or splitting is possible.

A number of variations on this basic theme are possible [13]. For example, one possibility is initially to split the image into a set of squares blocks. Further splitting is carried out as above, but merging is initially limited to groups of four blocks that are descendants in the quadtree representation and that satisfy the predicate $P$. When no further mergings of this type are possible, the procedure is terminated by one final merging of regions satisfying Step 2 above. At this point, the regions that are merged may be of different sizes. The principal advantage of this approach is that it uses the same quadtree for splitting and merging, until the final merging step.

An illustration of the split-and-merge algorithm discussed above is shown in Figure 6. The image under consideration consists of a single object and background. For simplicity, assume that both the object and background have constant gray levels and that $P(R_i) = \text{TRUE}$ if all pixels in $R_i$ have the same intensity. Then, for the entire image region $R$, it follows that $P(R) = \text{FALSE}$, so the image in split as shown in Figure 6.(a). In the next step, only the top left region satisfies the predicate so it is not changed, while the other three quadrant regions are split into subquadrants, as shown in Figure 6.(b). At this point several regions can be merged, with the exception of the two subquadrants that include the lower part of the object; these do not satisfy the predicate and must be split further. The result of the split-and-merge operation is shown in Figure 6.(c). At this point all regions satisfy $P$, and merging the

FIGURE 5
(a) Quadtree of region splitting; (b) Geographical position of each symbol
appropriate regions from the last split operation yields the final, segmented result shown in Figure 6.(d).

4 EXAMPLES

Two examples are introduced in this section to demonstrate the proposed approach. Both image segmentation methods introduced above were employed and the segmentation results are the same, but region growing is more time consuming than the region splitting and merging method. Hence, to save space, in this section, only region splitting and merging method will be used to demonstrate the feasibility of applying image segmentation methods in identification of hybrid CA.

4.1 One Dimensional Hybrid CA

Consider a one-dimensional CA with hybrid transition rules on a 300 × 300 lattice. This example includes three rules whose distribution is based on Figure 1.(a). Table 4.1 shows the definition of each rule, further details of which are given in [14]. In Table 4.1, \([-1, 0, 1]\) denotes neighbourhood
TABLE 1
The rule definitions for Example 1

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Neighbourhood</th>
<th>Rule Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>RuleA</td>
<td>{-1, 0, 1}</td>
<td>R60</td>
</tr>
<tr>
<td>RuleB</td>
<td>{-1, 0, 1}</td>
<td>R109</td>
</tr>
<tr>
<td>RuleC</td>
<td>{-2, -1, 0, 1, 2}</td>
<td>T209</td>
</tr>
</tbody>
</table>

FIGURE 7
Observed pattern of the hybrid one-dimensional CA for Example 1

c(j − 1; t), c(j; t), c(j + 1; t). The generated pattern by this rule is shown in Figure 7.

Because of the blurring of the boundaries between the two rules, the generated pattern is more complex than the traditional one-dimensional CA pattern. If traditional identification methods for uniform CA were applied to this system, it would be inevitably obtain a wrong model.

The new method presented in this paper was used to detect the rule distribution of the observed pattern. Initially, the pattern was mapped into a virtual image by selecting an appropriate \( n_b \). The value of each pixel was set using the calculated property of the corresponding block by neighbourhood detection and case extraction. The rule distribution was then detected using the method described above in Section 2 with different \( n_b \) and \( T_u \) and is shown in Figure 8.(a)-(e), which were generated by overlapping the detected rule distribution graph on the observed pattern. One rule is illustrated by one colour and the areas with no colour background indicate the unidentifiable regions. The polynomial model for each region was then generated using OLS, but the final expressions have been omitted in this paper to save space. The reconstructed
FIGURE 8
The rule distribution graph (a) and reconstructed patterns of Example 1 using different $T_u$ and $n_b$.
(a) $n_b = 30$, $T_u = 0.9$; (b) $n_b = 30$, $T_u = 0.8$; (c) $n_b = 20$, $T_u = 0.9$; (d) $n_b = 6$, $T_u = 0.9$; (e) $n_b = 30$, $T_u = 0.6$

patterns using the identified hybrid models with the same initial condition as the observed pattern are shown in Figure 8.(f)-(j) respectively. Figure 8.(a)-(c) clearly show that the presented approach detected three rules which are separated by some unidentifiable regions, most of which exist on the regions with mixed rules. Following a decreasing of $n_b$, the identifiable regions increase through comparing Figure 8.(a) and (c), but if too small an $n_b$ value is used, a wrong model may be obtained in some regions, as shown in Figure 8.(d). The right region of Figure 8.(d), which should have been detected as RuleC, was mistakenly detected as a mix rule because RuleC has a larger neighbourhood than that of RuleA and RuleB. RuleC has $2^5 = 32$ potential cases, which makes the possibility of getting an over-estimated result larger comparing with the $2^3 = 8$ potential cases of RuleA and RuleB. Following a decrease in $T_u$, the identified regions increase as well, which can be seen from a comparison between Figure 8.(a) and (b), but if too small a $T_u$ value is used, some regions with low SNR (signal noise ratio) would be identified as an incorrect model, as shown on the right part of Figure 8.(e) and (j). Experience suggests that for a hybrid CA in a $300 \times 300$ lattice, when $n_b$ is chosen between 5 and 20, and $T_u$ is chosen between 0.75 and 0.95, the algorithm always appears to produce a satisfactory segmentation, but an exception to this may occur if the rule has a large neighbourhood, which would imply more data should be collected and $n_b$ should be chosen to be larger.

Consider Figure 8.(a) where $n_b = 10$ and $T_u = 0.9$. The detected neighbourhood and properties of the regions illustrated by different colours are shown in Table 4.1. Comparison between Table 4.1 with Table 4.1 and a comparison between Figure 1.(a) and Figure 8.(f) clearly shows that the regions with coloured backgrounds are segmented correctly. Although some regions are marked as unidentifiable, this is reasonable because a real system is always likely to be corrupted by noise in some regions, which could not be identified.
Moreover, in some systems, only parts of rules are of interest, hence it is not necessary to detect all regions.

### 4.2 Two Dimensional Hybrid Spatio-Temporal system

Excitable media systems are usually simulated in the computer using a CA model [9]. Therefore, the identification of excitable media using CA models is attracting more and more investigations [22]. A challenge which has never been tried before is to try and identify a hybrid spatio-temporal system combining an excitable media rule and the traditional CA rule.

This section introduces a two-dimensional hybrid spatio-temporal example on a 240 \times 240 lattice, whose rule distribution is illustrated in Figure 9.(a), where the blue part is an excitable media system and the green part is a traditional CA called *Conway’s game of life* with a Moore neighbourhood.

The parameters of the excitable media system were set as: \( E = 5, R = 8, T = 3 \), with a Moore neighbourhood (further details of the generation of excitable media systems can be seen in [22]). With random initial data, 70 time frames were generated, and three of them are shown in Figure 9.(b)-(d).

Using the proposed algorithm, the rule distributions were detected and are shown in Figure 10.(a)-(e) with \( T_u = 0.9 \) and different \( n_b \). Figure 10.(a)-(e) clearly shows that the algorithm detected two different rules, represented by the blue and green colours respectively. Following the decreasing of \( n_b \), the regions were partitioned more precisely. However when \( n_b \) decreased to 3, the algorithm introduced some nonexistent rules shown in Figure 10.(d).

![FIGURE 9](image)

*FIGURE 9*

The rule distribution image and three frames of generated hybrid CA for Example 2

<table>
<thead>
<tr>
<th>Colour</th>
<th>Neighbourhood</th>
<th>Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Red</td>
<td>{-1, 0, 1}</td>
<td>60</td>
</tr>
<tr>
<td>Green</td>
<td>{-1, 0, 1}</td>
<td>109</td>
</tr>
<tr>
<td>blue</td>
<td>{-2, -1, 0, 1, 2}</td>
<td>17532518</td>
</tr>
</tbody>
</table>

**TABLE 2**

The detected rule description for Example 1
and (e), which could be viewed as noise. This seems to occur because the number of measurements is small compared to the number of possible discrete states due to the small $n_0$, which results in over-estimation in some blocks. Apparently, one solution to this problem is to avoid selecting too small a value of $n_0$, which depends on the size of the lattice and the number of sampled frames. Another solution is to employ a well known filter in image processing named median filtering [11], which replaces the considered pixel value by the median of the values in a neighbourhood of that pixel. This method is particularly effective when the noise pattern consists of strong, spike-like components. Figure 10.(d) was processed by median filtering and the result is illustrated by Figure 10.(i), which clearly shows that all the noise has been excluded and the detected image is closer to Figure 9.(a), the original rule distribution image. However, Figure 10.(j), the filtered image of 10.(e), still has a bit of noise even after processing by median filtering because of lots of noise. Therefore, the combination of the selection of $n_0$ and median filtering is recommended in this paper to achieve the best segmentation result. In this example, $n_0 = 3, T_u = 0.9$ together with median filtering was the final parameters setting.

5 CONCLUSIONS

The procedure for the identification of hybrid CA can be summarized by three steps:(1) Map the CA pattern into a virtual image, the key of which is to calculate the property of each pixel by detecting the neighbourhood and extracting the cases; (2) Segment the virtual image using Region Growing or
Region Splitting and Merging Method: (3) Generate the polynomial model for each rule using Orthogonal Least Squares (OLS)[15]. As the creative and most difficult step, the first step sets up the relationship between the realistic CA pattern and the virtual image by introducing two parameters: $n_b$, which defines how many cells of the CA pattern are included in one pixel of the image; $T_u$, which represents the noise-immune ability. The smaller $n_b$ is, the more precise the segmentation would be, but which will more probability result in an over-estimated final result. To overcome this problem, the combination of the selection of $n_b$ and median filtering is recommended to achieve the best segmentation result. The smaller $T_u$ is, the more probability a block could be identified, but there is more risk of obtaining the wrong rule. A range of $T_u$ from experience, has been provided in this paper. Both Region Growing and Region Splitting and Merging Methods are popular methods in image segmentation, but Region Growing needs the manual selection of a set of seeds if all rules over the system are to be detected. If the identification is only for a special rule, it may be more effective to use Region Growing by initially selecting a seed from this rule and using the algorithm to find the boundary to separate this rule from the other rules. Two examples, using Region Splitting and Merging Methods have been employed in this paper and an inspection of the results shows that hybrid rules can be identified almost perfectly if an appropriate $n_b$ and $T_u$ can be selected.

Hybrid cellular automata can of course be more complex than the example described in this paper. For example, the rule could be changed in the same spatial position during the evolution. This paper presents preliminary results and further investigations are required to deal with the more complex cases which could occur in real systems.

REFERENCES


