Syntactic Generation of Similar Pictures

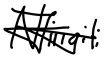
Nuru Jingili

A thesis submitted to the Faculty of Science,
University of the Witwatersrand, Johannesburg,
in fulfilment of the requirements for the Degree of
Doctor of Philosophy in Computer Science

November 11, 2019

Declaration

I, Nuru Mohamed Jingili, hereby declare that the work submitted is my own, except where work which has formed part of jointly-authored publications has been included. I confirm that appropriate credit has been given within this thesis where reference has been made to the work of others. This thesis is submitted for the degree of Doctor of Philosophy at the University of the Witwatersrand, Johannesburg. It has not been submitted before for any degree or examination at any other university.

Signature of candidate: 

Signed on November 11, 2019 in Johannesburg.

Abstract

Syntactic picture generation is a significant field in mathematics and computer science introduced over three decades ago. Currently, there is interest in research on the syntactic generation of similar pictures as a result of applications that require the use of similar pictures such as visual password systems. However, the problem with many syntactic picture generators currently available is that they generate an infinite number of pictures without considering the similarity between pictures. Addressing this challenge requires methods that can generate in a controlled manner a finite number of pictures, which are similar.

This study presents methods of generating in a controlled manner a finite number of pictures that are similar using bag context picture grammars and random context picture grammars. Firstly, we introduce bag context picture grammars and show how we can generate pictures using them. Next, we present three methods of generating in a controlled way a finite number of pictures that are similar by using both bag context picture grammars and random context picture grammars. The first method is limiting the number of refinements (subdividing of a picture into smaller sub-pictures) of pictures that can be produced by a given picture grammar. For this, all pictures generated by the grammar had the same number of refinements while the distribution of colours varied. The second method is limiting the number of refinements of selected sub-pictures of the picture. The last method is a combination of limiting the number of refinements of pictures and selected sub-pictures of the picture. Furthermore, we compared the simplicity of using both bag and random context picture grammars in generating similar pictures. Moreover, we used a mathematical similarity measure (in this case the spatial colour distribution descriptor) to evaluate the performance of the presented methods and the outcome verified that there is an improvement in the similarity of the generated pictures when using the presented methods. We then collected human perceptions on the similarity of generated pictures in an online survey. The results of the survey were used to evaluate whether there is a correlation between human perceptions and the spatial colour distribution descriptor. The results showed that there is a correlation.

Publications

1. Jingili, N., Ewert, S., and Sanders, I. D. (2018). Measuring perceptual similarity of syntactically generated pictures. *Proceedings of 8th International Conference on Simulation and Modeling Methodologies, Technologies and Applications, SIMULTECH 2018* Porto, Portugal, July 29-31, 2018., pages 244-255. SciTePress.

2. Ewert, S., Jingili, N., Mpota, L., and Sanders, I. D. (2019). Bag context picture grammars, Journal of Computer Languages, Volume 51, 2019, Pages 214-221,

3. Jingili, N., Ewert, S., and Sanders, I. D. (2018). Syntactic generation of similar pictures. Submitted to the *Advances in Intelligent Systems and Computing Series book (Springer)*, 2018.

4. Jingili, N., Ewert, S., and Sanders, I. D. (2019). Random and bag context picture generators. Submitted to the *South African Computer Journal*, 2019.

Acknowledgement

I would like to thank my supervisors Prof. Sigrid Ewert and Prof. Ian Sanders for accepting me as their student, for introducing me to this field and their guidance, support, encouragement and patience throughout my research. I am very grateful to Prof. Sigrid Ewert for funding my attendance at the SIMULTECH 2018 conference. I would also like to acknowledge my supervisors for co-authored work in SIMULTECH 2018, and other work that is submitted.

I would also like to acknowledge the help of Marike Kluyts in writing my papers and thesis. She has availed time to read and guide me in proper academic writing.

I want to thank and acknowledge the support of colleagues, Liemiso Mpota, Abejide Ade-Ibijola, Blessing Ogbuokiri, Kehinde Arubela and Olaperi Okuboyejo who at different times have played a part in making my life easier at Wits.

I want to thank Mohsin Desai, Kgomotso Monyepote, Kebadiretse Mosiane and Precious Shabalala for their support in different administrative issues.

I would also like to acknowledge different organisations who have provided me with funding for this research as follows:

- CSIR DST scholarship for the year 2016 to 2018
- DST-NRF CoE-MaSS doctoral bursary for the year 2016
- University of the Witwatersrand, Johannesburg, postgraduate merit award (PMA) for the year 2018

Contents

1 Introduction **1**

1.1 Overview 1
1.2 Problem Statement 3
1.3 Research Objectives 3
1.4 Research Methodology 4
1.5 Contribution of the Research 5
1.6 Structure of the Document 5

2 Literature Review **7**

2.1 Picture Grammars 7
2.1.1 Notation 8
2.1.2 Random Context Picture Grammars 8
2.1.3 Generalised Random Context Picture Grammars . . 14
2.1.4 Bag Context Tree Grammars 16
2.1.5 Lindenmayer Systems 19
2.1.6 Array Grammars 23
2.1.7 Puzzle Grammars 25
2.1.8 Iterated Function Systems 27
2.1.9 Collage Grammars 29
2.1.10 Shape Grammars 31
2.1.11 Summary of Picture Grammar Types 32
2.2 Mathematical Similarity Measures 34
2.2.1 Texture Descriptors 34
2.2.2 Colour Descriptors 35
2.2.3 Tree Edit Distance (TED) 35
2.2.4 Summary of Mathematical Similarity Measures . . . 36
2.3 Perceptual Similarity 36
2.4 Visual Password Systems 38
2.4.1 Recall-Based Visual Password Systems 38
2.4.2 Recognition-Based Visual Password Systems 40
2.4.3 Cued-Recall-Based Visual Password Systems 41
2.4.4 Summary of Visual Password Systems 41
2.5 Conclusion 46

3 Bag Context Picture Grammars **47**
3.1 Introduction . . . 47
3.2 Definitions . . . 48
3.2.1 Bag Context Picture Grammars . . . 48
3.3 Rewriting an RCPG as a BCPG . . . 55
3.3.1 Notation . . . 55
3.3.2 The Special Case of RPCPGs . . . 58
3.3.3 The Special Case of RFCPGs . . . 59
3.3.4 The Special Case of CFPGs . . . 61
3.3.5 Summary of Rewriting an RCPGs as a BCPGs . . . 63
3.4 The Power of Bag Context Picture Grammars . . . 63
3.5 Conclusion . . . 76

4 Generation of Similar Pictures using BCPGs and RCPGs **78**
4.1 Introduction . . . 78
4.2 Generation of Similar Pictures . . . 79
4.2.1 Limiting the Refinement Levels . . . 79
4.2.2 Limiting Refinement of Sub-Pictures . . . 95
4.2.3 Limiting Both Refinement Levels and Refinement of Sub-Pictures . . . 105
4.3 Review of the Approaches of Generation of Similar Pictures 119
4.3.1 Limiting Refinement Levels . . . 119
4.3.2 Limiting Refinement of Sub-Pictures . . . 119
4.3.3 Limiting Both Refinement Levels and Refinement of Sub-Pictures . . . 119
4.4 Conclusion . . . 120

5 Estimating Similarity Mathematically **121**
5.1 Introduction . . . 121
5.2 Mathematical Similarity Measure . . . 122
5.3 Determining the Similarity of Pictures . . . 122
5.4 Results . . . 123
5.4.1 Limiting the Number of Refinements . . . 135
5.4.2 Limiting Refinement of Sub-Pictures . . . 138
5.4.3 Limiting Both Refinement Levels and Refinement of Sub-Pictures . . . 140
5.5 Discussion . . . 143
5.6 Conclusion . . . 144

6 Perceptual similarity **146**
6.1 Introduction . . . 146
6.2 Results . . . 147
6.2.1 Ranking of Pictures in Gallery . . . 148

6.2.2 Similarity of Pictures in Gallery 158
6.2.3 Ranking of Galleries 159
6.2.4 Factors that Determine Similarity 160
6.3 Evaluation 160
6.4 Conclusion 163

7 Conclusion **164**

References **166**

Appendix **175**

A Ethics Clearance **176**

B Sample Online Survey **178**

C Syntactic Picture Generators **192**
C.1 Introduction 192
C.2 Design Considerations 193
C.2.1 Assumptions 193
C.2.2 Software Development Methods 193
C.2.3 System Environment 193
C.2.4 User Interface 193
C.2.5 Integrated Development Environment (IDE) 193
C.2.6 Constraints 194
C.3 BCPG Picture Generator 194
C.3.1 System Design 194
C.3.2 BACOPA Control Flow 194
C.3.3 System Output 194
C.4 RCPG Picture Generator 195
C.4.1 System Design 195
C.4.2 RACOPA Control Flow 198
C.4.3 System Output 198
C.5 Pictures Generated by BACOPA and RACOPA 199
C.6 Future Work 201

D Visual Password Systems **202**
D.1 Design of the Visual Password System 202
D.2 Conclusion 204

List of Figures

2.1 Production in RCPG 9
2.2 Some pictures generated by G_{lights}. 11
2.3 Rules for grammar G_{lights} 12
2.4 $\Psi_G(\{b\})$ is a dark triangle 15
2.5 $\Psi_G(\{b\})$ is a dark square 16
2.6 Rules for grammar G_{gasket} 16
2.7 Rules for grammar G_{tree} 18
2.8 An example of a tree generated by G_{tree} 18
2.9 Some pictures generated by Stochastic L-systems [69] 21
2.10 ETOL picture sequence [16] 22
2.11 Variety of pictures by an ETOL chain-code picture language [16] 22
2.12 Stochastic L-system rules 22
2.13 The first ETOL-system table 22
2.14 The second ETOL-system table 23
2.15 Production rules of G_{array} 24
2.16 A picture generated by G_{array} 25
2.17 A basic puzzle grammar production rules 26
2.18 A picture generated by G_{puzzle} 27
2.19 Production rules for the grammar G_{puzzle} 28
2.20 Some pictures generated by $G_{\text{triangles}}$ 28
2.21 Example of pictures produced by MRFS 29
2.22 Production rules of G_{collage} 30
2.23 A derivation in G_{collage} 30
2.24 Some pictures generated by G_{collage} 31
2.25 Example of a shape grammar 32
2.26 Application of the shape grammar in Figure 2.25 33
2.27 Example of input of DAS password on a 4×4 grid 39
2.28 Example of Passfaces system on a 3×3 grid containing users password and distractor images 42
2.29 Example of first authentication step 43
2.30 Example of second authentication step 44
2.31 Example of third authentication step 44

2.32 Example of fourth authentication step 45
2.33 Example of user Passpoint password with click order displayed 45

3.1 Production in BCPG 49
3.2 A selection of pictures in $\mathcal{G}_{\text{carpet}}$ 52
3.3 Rules for grammar G_{carpet} 52
3.4 A selection of pictures in $\mathcal{G}_{\text{random}}$ 57
3.5 Rules for grammar G_{random} 57
3.6 Rules for grammar $G_{\text{random:bag}}$ 58
3.7 Some pictures in $\mathcal{G}_{\text{permit}}$. 60
3.8 Rules for grammar G_{permit} 60
3.9 Rules for grammar $G_{\text{permit:bag}}$ 61
3.10 Rules for grammar G_{forbid} 62
3.11 Rules for grammar $G_{\text{forbid:bag}}$ 62
3.12 A selection of pictures in $\mathcal{G}_{\text{free}}$ 64
3.13 Rules for grammar G_{free} 64
3.14 Rules for grammar $G_{\text{free:bag}}$ 64
3.15 Some pictures in $\mathcal{G}_{\text{random:var}}$ 68
3.16 Rules for grammar $G_{\text{random:bag:var}}$ 69
3.17 Coloured rules for grammar $G_{\text{random:var}}$ 70
3.18 Some pictures in $\mathcal{G}_{\text{permit:varA}}$ 72
3.19 Set of rules for grammar $G_{\text{permit:bag:varA}}$ 72
3.20 Set of rules for grammar $G_{\text{permit:varA}}$ 73
3.21 Some pictures in $\mathcal{G}_{\text{permit:varB}}$ 74
3.22 Set of rules for grammar $G_{\text{permit:bag:varB}}$ 75
3.23 Set of rules for grammar $G_{\text{permit:varB}}$ 76

4.1 A selection of pictures in $\mathcal{G}_{\text{circlesMixed}}$ 81
4.2 Rules for grammar $G_{\text{circlesMixed:bag}}$ 82
4.3 Rules for grammar $G_{\text{circlesMixed}}$ 83
4.4 Some pictures in $\mathcal{G}_{\text{circlesFirst}}$ 85
4.5 Rules for grammar $G_{\text{circlesFirst:bag}}$ 86
4.6 Rules for grammar $G_{\text{circlesFirst}}$ 87
4.7 Some pictures in $\mathcal{G}_{\text{circlesSecond}}$ 88
4.8 Rules for grammar $G_{\text{circlesSecond:bag}}$ 89
4.9 Rules for grammar $G_{\text{circlesSecond}}$ 90
4.10 Some pictures in $\mathcal{G}_{\text{circlesThird}}$ 92
4.11 Rules for grammar $G_{\text{circlesThird:bag}}$ 93
4.12 Rules for grammar $G_{\text{circlesThird}}$ 94
4.13 Some pictures in $\mathcal{G}_{\text{flowerMixed}}$ 96
4.14 Rules for grammar $G_{\text{flowerMixed:bag}}$ 97
4.15 Rules for grammar $G_{\text{flowerMixed}}$ 98
4.16 Some pictures in $\mathcal{G}_{\text{flowerThree}}$ 100

4.17 Rules for grammar $G_{\text{flowerThree:bag}}$ . . . 101
4.18 Rules for grammar $G_{\text{flowerThree}}$ . . . 102
4.19 Some pictures in $\mathcal{G}_{\text{flowerTwo}}$ . . . 103
4.20 Rules for grammar $G_{\text{flowerTwo:bag}}$ . . . 104
4.21 Rules for grammar $G_{\text{flowerTwo}}$ . . . 105
4.22 Some pictures in $\mathcal{G}_{\text{carpetMixed}}$ . . . 106
4.23 Rules for grammar $G_{\text{carpetMixed:bag}}$ . . . 108
4.24 Rules for grammar $G_{\text{carpetMixed}}$ . . . 109
4.25 Some pictures in $\mathcal{G}_{\text{carpetSecond}}$ . . . 110
4.26 Rules for grammar $G_{\text{carpetSecond:bag}}$ . . . 111
4.27 Rules for grammar $G_{\text{carpetSecond}}$ . . . 112
4.28 Some pictures in $\mathcal{G}_{\text{carpetThird}}$ . . . 113
4.29 Rules for grammar $G_{\text{carpetThird:bag}}$ . . . 114
4.30 Rules for grammar $G_{\text{carpetThird}}$ . . . 115
4.31 A selection of pictures in $\mathcal{G}_{\text{carpetThirdgrey}}$ . . . 116
4.32 Rules for grammar $G_{\text{carpetThirdgrey:bag}}$ . . . 117
4.33 Rules for grammar $G_{\text{carpetThirdgrey}}$ . . . 118

5.1 SpCD image retrieval procedure . . . 123
5.2 Sierpiński carpet, different refinements . . . 124
5.3 Sierpiński carpet, first refinement . . . 125
5.4 Sierpiński carpet, second refinement . . . 126
5.5 Sierpiński carpet, third refinement . . . 127
5.6 Sub-pictures refined up to three times . . . 128
5.7 Top right quarter and the bottom left quarter refined exactly thrice . . . 129
5.8 Top left quarter and the bottom right quarter subdivided exactly twice . . . 130
5.9 Different refinements with grey or black sub-pictures in the corners and centre . . . 131
5.10 Second refinement with grey or black sub-pictures in the corners and centre . . . 132
5.11 Third refinement with grey or black sub-pictures in the corners and centre . . . 133
5.12 Third refinement with grey sub-pictures in the corners and centre . . . 134

6.1 Gallery A: Sierpiński carpet, different refinements . . . 148
6.2 Gallery B: Sierpiński carpet, first refinement . . . 149
6.3 Gallery C: Sierpiński carpet, second refinement . . . 150
6.4 Gallery D: Sierpiński carpet, third refinement . . . 151
6.5 Gallery E: Flowers . . . 152
6.6 Gallery F: Flowers . . . 153

6.7 Gallery G: Flowers 154

C.1 The initial bag and grammar rules as they appear in `bcgrammar.py` file 195
C.2 Activity diagram: bag context picture generator 196
C.3 An example of the output of BACOPA application 197
C.4 The grammar rules as they appear in `rcgrammar.py` file . . 197
C.5 Activity diagram: random context picture generator 198
C.6 An example of the output of the RACOPA application . . . 199
C.7 Some pictures generated by BACOPA and RACOPA 200

D.1 Visual password system workflow 203
D.2 Example of the first password selection screen 205
D.3 Example of second password selection screen 206
D.4 Example of third password selection screen 207
D.5 Example of fourth password selection screen 208

List of Tables

2.1 Derivation of Figure 2.2a 12
2.2 The evolution of the bag in a derivation of Figure 2.8 19

3.1 Derivation of Figure 3.2a 53

5.1 Similarity of Figure 5.2a to pictures in Figure 5.2 135
5.2 Similarity of Figure 5.2d to pictures in Figure 5.2 136
5.3 Similarity of Figure 5.2f to pictures in Figure 5.2 136
5.4 Similarity of Figure 5.3a to pictures in Figure 5.3 137
5.5 Similarity of Figure 5.4a to pictures in Figure 5.4 137
5.6 Similarity of Figure 5.5a to pictures in Figure 5.5 138
5.7 Similarity of Figure 5.6h to pictures in Figure 5.6 139
5.8 Similarity of Figure 5.6c to pictures in Figure 5.6 139
5.9 Similarity of Figure 5.7a to pictures in Figure 5.7 139
5.10 Similarity of Figure 5.8a to pictures in Figure 5.8 140
5.11 Similarity of Figure 5.9c to pictures in Figure 5.9 141
5.12 Similarity of Figure 5.10c to pictures in Figure 5.10 141
5.13 Similarity of Figure 5.11c to pictures in Figure 5.11 142
5.14 Similarity of Figure 5.12c to pictures in Figure 5.12 142

6.1 Similarity of Figure 6.1c to pictures in Figure 6.1 155
6.2 Similarity of Figure 6.2c to pictures in Figure 6.2 156
6.3 Similarity of Figure 6.3c to pictures in Figure 6.3 156
6.4 Similarity of Figure 6.4c to pictures in Figure 6.4 157
6.5 Similarity of Figure 6.5c to pictures in Figure 6.5 157
6.6 Similarity of Figure 6.6c to pictures in Figure 6.6 158
6.7 Similarity of Figure 6.7c to pictures in Figure 6.7 158
6.8 Perception of similarity of pictures in each gallery 159
6.9 Ranking of galleries in Figures 6.1–6.4 according to similarity of pictures in gallery 160
6.10 Ranking of galleries in Figures 6.5–6.7 according to similarity of pictures in gallery 160
6.11 Most important factor when determining similarity of pictures 161
6.12 DCG calculation for Table 6.1 162

6.13 NDCG results 162

Chapter 1

Introduction

1.1 Overview

Pictures are widely used in our everyday life, hence they have become an essential part of many aspects of life such as creating art, book illustrations, printed media, video games, simulators, generating cloth, medical visualisation, prototype rendering, demonstrating mechanical processes and so on [14, 23, 92]. Syntactic picture generation was introduced more than three decades ago [23]. Since its introduction, there has been more research on it and new approaches have been recommended.

As the need to grammatically generate pictures has increased, the need for generating similar pictures has also become important. Many applications such as visual password systems (for example, Passfaces) [64] and modelling of plant objects [15] require the generation of similar pictures. This has created a rise of interest in research on the generation of similar pictures. Most picture grammars can be used to generate similar pictures. The problem with most popular grammatical picture generators currently available, however, is that they generate many pictures without considering the similarity between the pictures. To generate similar pictures efficiently, there is a need to determine effective methods of generating in a controlled way similar pictures.

Many picture-generating methods, both syntactic and non-syntactic, have been introduced in the literature. This research focused on syntactic picture generation particularly using bag context picture grammars (BCPGs) and random context picture grammars (RCPGs). We used BCPGs and RCPGs to generate in a controlled manner pictures which are structurally similar but not identical for use in visual password systems. We consider pictures to be structurally similar if they share some characteristics based on how they were generated, for example, the pictures contain the same sub-pictures or they have been refined the same number of times. We used the regulated rewriting in BCPGs and RCPGs to control the generation of structurally similar pictures. We present methods for generating in a

controlled manner a finite number of structurally similar pictures. For a given syntactically generated picture, the developed methods have the ability to generate structurally similar pictures.

This study developed BCPGs based on RCPGs and bag context tree grammars. In Chapter 3 we present the details of these grammars and show how we can generate pictures using them. BCPGs are context-free grammars, where the application of a rule is regulated by a k-tuple of integers, the bag, which changes during the derivation of a picture. A rule in the grammar can only be applied if the bag at that point is within the scope defined by the lower and upper limits of the bag for that rule. When a rule is applied, it causes the bag values to change by adding the bag adjustment, which is part of the rule. We then investigate some aspects of BCPGs. In particular, we compared RCPGs and BCPGs. We show that any RCPG can be written as a BCPG. We consider this process for three sub-classes of RCPGs and in each case, illustrate it with an example.

We then developed two picture-generating applications, using bag context picture grammars and random context picture grammars. Appendix C demonstrates how they work, their structure and functions. In essence, two grammar processing tools were developed and these are BACOPA (bag context picture application) and RACOPA (random context picture application) that process BCPGs and RCPGs respectively. BACOPA and RACOPA were mainly developed to simplify the processing of the grammars, to minimise errors when writing and modifying grammars and to provide quick visual results.

Next, we developed methods of generating in a controlled way a finite number of pictures that are structurally similar by using both bag context picture grammars and random context picture grammars. In Chapter 4 we present these methods using BCPGs and RCPGs. The first method is limiting the number of refinements of pictures that can be produced by a given picture grammar. For this, all pictures generated by the grammar had the same number of refinements, while the distribution of colours varied. The second method is limiting the number of refinements of selected sub-pictures of the picture. We then developed a method that uses both limiting refinement levels and refinement of sub-pictures. We also compared the simplicity of using bag versus random context picture grammars in generating similar pictures.

We then evaluated the performance of the presented methods by measuring the similarity of the produced pictures according to mathematical similarity measure (mathematical similarity) in Chapter 5. We used spatial colour distribution descriptor (SpCD) [6] to estimate mathematical similarity. The outcome verified that there is an improvement in the mathematical similarity of the generated pictures when using the presented methods.

Lastly, we collected human perceptions on the similarity of generated pictures in an online survey. In Chapter 6, we present the results of the survey and evaluate whether there is a correlation between how humans perceive similarity (perceptual similarity) and mathematical similarity. We compared the results of the survey with the results of the spatial colour distribution descriptor similarity measure on the same images. The results showed that there is a correlation between perceptual similarity and mathematical similarity. This implies that the spatial colour distribution descriptor can be used to judge the similarity of pictures generated by bag context picture grammars and random context picture grammars.

1.2 Problem Statement

Generation of similar pictures in a controlled way was motivated by previous research on the generation of similar pictures for visual password systems [64]. Many picture grammars can generate pictures that are similar but fail to generate in a controlled way only similar pictures together. For many of these grammars, to generate similar pictures, it requires the user to generate many pictures automatically and then manually select the pictures that are similar. Many picture grammars generate an infinite number of pictures without considering similarity; it is therefore, difficult to determine if the next generated picture will be similar to a selected picture.

Current research on the generation of similar pictures lacks guidance on how similar pictures can be generated in a controlled way. If pictures that are similar to a selected picture are needed, there should be ways of writing the grammar to generate only similar pictures. Okundaye et al. [63, 65] generated similar pictures using various picture grammars; however, the study failed to generate only similar pictures as some pictures in the galleries were quite different from the query picture. A query picture is a picture whose similarity is tested against all other pictures in a given picture set. Some of these grammars can be modified to generate a finite number of pictures that are structurally similar to a selected picture. This research used RCPGs and BCPGs to generate structurally similar pictures.

1.3 Research Objectives

The main objective of this research was to propose methods of generating in a controlled manner a finite number of pictures that are similar using RCPGs and BCPGs. The following is the list of primary objectives that our research accomplished:

- Defined bag context picture grammars and showed how we can generate

pictures using BCPGs.

- For BCPGs and RCPGs, we developed methods to generate structurally similar pictures for a given generated picture.
- Created a grammatical picture generator with the capability of generating pictures using BCPGs.
- Created a grammatical picture generator with the capability of generating pictures using RCPGs.
- Analysed the capability of the BCPGs and RCPGs to generate similar pictures using the developed methods by SpCD.
- Evaluated the correlation between perceptual similarity and mathematical similarity.

1.4 Research Methodology

This research was conducted as follows:

1. We first studied random context picture grammars and bag context tree grammars to learn how they work in order to define bag context picture grammars. We defined bag context picture grammars, and showed how to rewrite RCPGs as BCPGs.
2. Then we developed two picture-generating applications which were used in this research to generate pictures using BCPGs and RCPGs (Appendix C).
3. Afterwards, we developed methods to generate structurally similar pictures using BCPGs and RCPGs. We also showed it is easier to use BCPGs than RCPGs to generate structurally similar pictures. Easier grammar to use in this context means a grammar type that requires few rules and variables to write, and in case there is a change in a few attributes of the gallery, there should be minimal need for changing the rules.
4. SpCD was used to determine the similarity of the generated pictures. The results of SpCD were then used to determine whether BCPGs and RCPGs can generate mathematically similar pictures using the presented methods.
5. An online survey was conducted on SurveyMonkey to determine perceptual similarity of the pictures generated by BCPGs and RCPGs. The results of the online survey were then compared to spatial colour

distribution descriptor similarity values to determine if perceptual similarity is consistent with mathematical similarity.

1.5 Contribution of the Research

The main contribution of this research is the developed methods for generating structurally similar pictures using BCPGs and RCPGs. Although picture grammars have been used before to generate similar pictures, they lacked methods for generating only pictures similar to a given picture. This research has explored the power of regulated rewriting in BCPGs and RCPGs to generate structurally similar pictures. To the best of the author's knowledge, this is the first time BCPGs and RCPGs have been used to generate structurally similar pictures by modifying the grammar.

Another contribution of this research is the introduction of bag context picture grammars. We have also provided a comparison of BCPGs and RCPGs in the generation of structurally similar pictures by modifying the grammar. This comparison is significant as it recommends the grammar type which is easy to use for the generation of structurally similar pictures in this way. Furthermore, this research recommends a mathematical similarity measure to be used for analysing similarity of syntactically generated pictures, particularly BCPGs and RCPGs.

1.6 Structure of the Document

The remainder of this thesis is organised as follows. Chapter 2 presents background information on picture grammars, mathematical similarity, perceptual similarity and visual password systems. Chapter 3 presents the definitions of BCPGs, RCPGs and all relevant concepts. We show that any RCPG can be written as a BCPG. We consider this process for three sub-classes of RCPGs and, in each case, illustrate it with an example. Moreover, we consider three sets of pictures and discuss why it is easier to write a BCPG that generates a variation on the set, and modify that BCPG again, should we wish to, than to do the same to the RCPG that generates the original picture set. In Chapter 4, we present the methods we used in generating structurally similar pictures using BCPGs. Moreover, we investigate which grammar type among RCPGs and BCPGs is the easiest to use in generating galleries of structurally similar pictures. In Chapter 5, we determine the mathematical similarity of the generated pictures, using spatial colour distribution descriptor and evaluate the effectiveness of the developed methods in generating mathematically similar pictures. Chapter 6 presents the results of an online survey that we conducted to determine how

humans perceive the similarity of our generated pictures. We then evaluate the relationship between perceptual similarity and mathematical similarity results. Chapter 7 presents the conclusion and future work.

Chapter 2

Literature Review

This chapter presents different syntactic picture generating mechanisms that have demonstrated the current problems with the picture grammars currently available. We investigated a variety of picture grammars in order to learn their capability on the generation of pictures and their potential for developing new picture grammar types. We also discuss several mathematical similarity measures for measuring the mathematical similarity of pictures based on texture, colour and tree edit distance. We then look at what different researchers have done regarding perceptual similarity. Lastly, we discuss previous research on visual password systems. In this, we look at what visual password systems are, why they are gaining popularity, what different types of visual password systems are, and why pictures generated by formal grammars are ideal.

2.1 Picture Grammars

During the last decades, several methods of syntactic picture generation have been introduced. These picture grammar types range from context-free grammars, such as chain code picture grammars [35, 36], which cannot generate a square, to basic puzzle grammars that can generate isosceles right triangles [84]. A more recent development is the concept of cooperating context-free array grammar systems that use permitting features [86]. These systems can generate all hollow square frames in the so-called terminal mode, using only two array grammars.

For our research, we studied different methods of syntactic picture generation. It was vital for us to examine different grammar types and their picture generating mechanisms to determine their capability. While there are many picture grammar types for generating pictures, few grammars provide easy control of the application of rules when generating pictures. The problem with many picture grammar types available is that some of them are context-free which makes them easy to use but weak for many

real-life problems, while others are context-sensitive, although strong, they are sometimes very restrictive and difficult to use. As this research is based on the generation in a controlled way of a finite number of structurally similar pictures, we need intermediate picture grammar types for generating them. These grammars should have the simplicity of context-free grammars and the strength of context-sensitive ones. An example of such a grammar type is random context picture grammar. The introduction of bag context tree grammars (BCTGs) by Drewes et al. [18], which are also context-free with regulated rewriting, has motivated the development of bag context picture grammars, which are discussed on Chapter 3.

We investigate the following grammar types in connection to their generation of pictures and potential to develop grammar types that can be used in the generation of pictures; random context picture grammars, bag context tree grammars, iterated function systems, L-systems, and array picture grammars and provide references of other picture grammar types.

We first present the list of notations that will be required throughout the thesis [18, 24, 92].

2.1.1 Notation

Let $\mathbb{N} = \{0, 1, 2, \ldots\}$, $\mathbb{N}_+ = \{1, 2, \ldots\}$ and $\mathbb{Z} = \{\ldots, -2, -1, 0, 1, 2, \ldots\}$. The sets $\mathbb{N} \cup \{\infty\}$ and $\mathbb{Z} \cup \{-\infty, \infty\}$ are denoted by $\mathbb{N}_\infty$ and $\mathbb{Z}_\infty$, respectively. Moreover, for $m \in \mathbb{N}_+$, let $[m] = \{1, 2, \ldots, m\}$.

If $I = \{1, \ldots, k\}$, then elements of $\mathbb{Z}_\infty^I$ are written as k-tuples. On $\mathbb{Z}_\infty^I$, addition (+), subtraction (-), and scalar multiplication ($\cdot$) are defined component-wise in the usual way. Similarly, $\leq$ denotes component-wise ordering. An element q of $\mathbb{Z}$ which occurs in the place of a vector, denotes the vector of the appropriate size with all components equal to q. Finally, for $\beta \in \mathbb{Z}^I$ and any $A \in I$, the A-component of β is denoted β_A or $\beta(A)$.

A signature is a set Σ of ranked symbols $f^{(k)}$ (where $k \geq 0$ is the rank). The set T_Σ of all trees over Σ is defined as usual: it is the smallest set such that $a \in T_\Sigma$ for all $a^{(0)} \in \Sigma$, and $f[t_1, \ldots, t_k] \in T_\Sigma$ for all $f^{(k)} \in \Sigma (k \geq 1)$ and $t_1, \ldots, t_k \in T_\Sigma$. The set of trees over $\Sigma \cup N$ is denoted by $T_{\Sigma \cup N}$.

2.1.2 Random Context Picture Grammars

One method of syntactic picture generation is random context picture grammars [24, 92]. These are context-free grammars with regulated rewriting, where the application of a rule in the grammar is regulated by two sets of variables, the so-called permitting ($\mathcal{P}$) and forbidding ($\mathcal{F}$) context sets. For these grammars, several theoretical results have been proved [1, 27–29, 39, 91, 92]. For example in [1], it was shown that random forbidding

context picture grammars are more powerful than table-driven context-free picture grammars. In [28], it was shown that random permitting context picture grammars are weaker that random context picture grammars. Moreover in [29], it was shown that grammars which use forbidding context only are strictly weaker than random context picture grammars.

RCPGs were developed with the aim of having a grammar class that is more powerful than context-free grammars and not difficult to use like context-sensitive grammars [24, 92]. The following are some subclasses of RCPGs which are of interest in this research: random permitting context picture grammars (RPCPGs), random forbidding context picture grammars (RFCPGs) and context-free picture grammars (CFPGs).

The use of context in RCPGs makes the grammar class an ideal candidate for the generation of a finite number of similar pictures as the context provides control in the application of the rules. RCPGs have been seen to generate galleries of pictures with variations. All definitions for RCPGs below were adapted from [23, 24, 92].

Random context picture grammars generate pictures using productions of the form in Figure 2.1, where A is a variable, x_{11}, $x_{12}, \ldots, x_{mm}$ are variables or terminals for $m \in \mathbb{N}_+$, and $\mathcal{P}$ and $\mathcal{F}$ are sets of variables. The interpretation is as follows: if a developing picture contains a square labelled A and if all variables of $\mathcal{P}$ and none of $\mathcal{F}$ appear as labels of squares in the picture, then the square labelled A may be divided into squares of equal size with labels $x_{11}, x_{12}, \ldots, x_{mm}$.

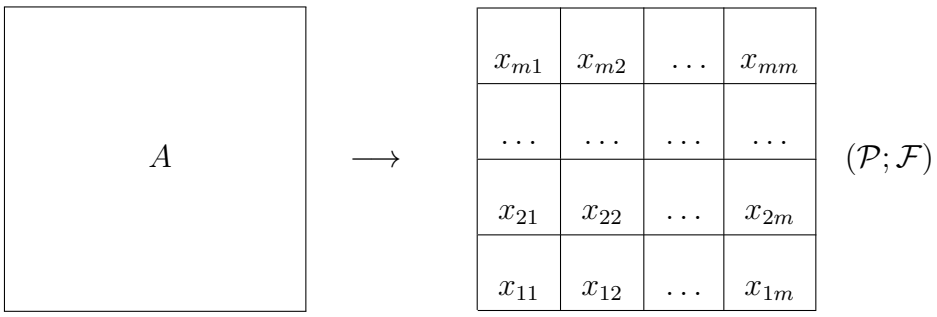


Figure 2.1: Production in RCPG

Definition 2.1. A random context picture grammar $G = (V_{\mathrm{N}}, V_{\mathrm{T}}, P, (S, \sigma))$ has a finite alphabet V of labels, consisting of disjoint subsets V_{N} of variables and V_{T} of terminals. P is a finite set of productions of the form $A \to [x_{11}, x_{12}, \ldots, x_{mm}](\mathcal{P}; \mathcal{F})$ with $m \in \mathbb{N}_+$, where $A \in V_{\mathrm{N}}$, $x_{11}, x_{12}, \ldots, x_{mm} \in V$ and $\mathcal{P}, \mathcal{F} \subseteq V_{\mathrm{N}}$. Finally, there is an initial labelled square (S, σ) with $S \in V_{\mathrm{N}}$.

Definition 2.2. If every production in G has $\mathcal{P} = \mathcal{F} = \emptyset$, we call G a context-free picture grammar (CFPG); if $\mathcal{F} = \emptyset$ for every production, G is a random permitting context picture grammar, and when $\mathcal{P} = \emptyset$, G is a random forbidding context picture grammar.

Definition 2.3. A pictorial form is any finite set of non-overlapping labelled squares in the plane. If Π is a pictorial form, the set of labels used in Π is denoted by $l(\Pi)$. The size of a pictorial form Π is the number of squares contained in it, denoted by $|\Pi|$.

For an RCPG G and pictorial forms Π and Γ, we write $\Pi \Longrightarrow \Gamma$ if there is a production $A \to [x_{11}, x_{12}, \ldots, x_{mm}](\mathcal{P}; \mathcal{F})$ in G, Π contains a labelled square (A, α), $l(\Pi \setminus \{(A, \alpha)\}) \supseteq \mathcal{P}$ and $l(\Pi \setminus \{(A, \alpha)\}) \cap \mathcal{F} = \emptyset$, and $\Gamma = (\Pi \setminus \{(A, \alpha)\}) \cup \{(x_{11}, \alpha_{11}), (x_{12}, \alpha_{12}), \ldots, (x_{mm}, \alpha_{mm})\}$. As above, $\Longrightarrow^*$ denotes the reflexive transitive closure of $\Longrightarrow$.

Definition 2.4. A picture is a pictorial form Π with $l(\Pi) \subseteq V_T$. A sub-picture Ω of a picture Π is any subset of Π that fills a square, that is, the union of all the squares in Ω is a square. The gallery $\mathcal{G}(G)$ generated by a grammar $G = (V_{\mathrm{N}}, V_{\mathrm{T}}, P, (S, \sigma))$ is the set of pictures Π such that $\{(S; \sigma)\} \Longrightarrow_G^* \Pi$. The galleries generated by RCPGs, CFPGs, RPCPGs and RFCPGs are called random context galleries (RCPLs), context-free galleries (CFPLs), random permitting context galleries (RPCPLs) and random forbidding context galleries (RFCPLs), respectively.

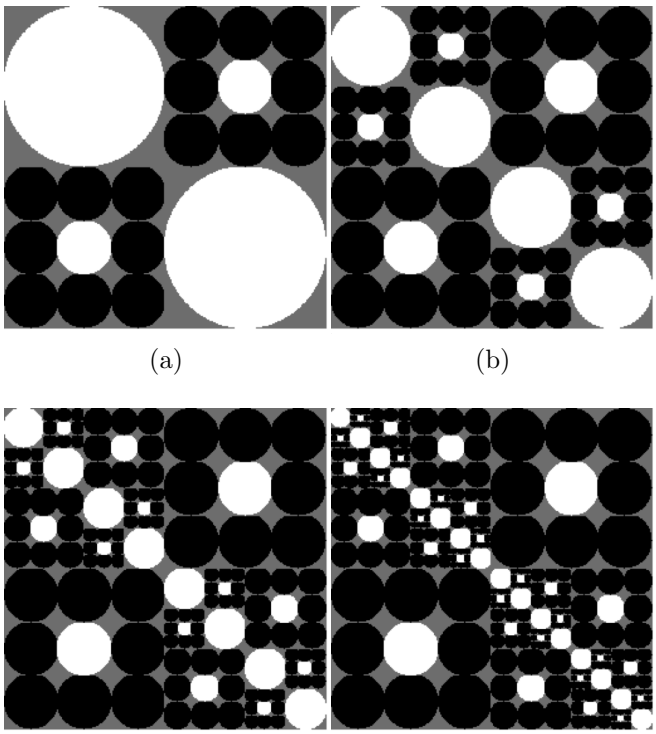
The following is an example of a gallery that is created by a random context picture grammar.

Example 2.1. Consider the gallery $\mathcal{G}_{\text{lights}}$ shown in Figure 2.2. The white circles in a grey background resemble lights in the dark. Four pictures in $\mathcal{G}_{\text{lights}}$ are shown in Figure 2.2, where Figure 2.2a is the smallest picture in this gallery, Figure 2.2b the second smallest, and so forth.

The smallest picture (Figure 2.2a) is divided into four equal sub-squares. The sub-squares in the top left and bottom right are filled with white circles on a grey background. The sub-squares in the bottom left and top right consist of the following image: in the centre, there is a white circle on a grey background, while the surrounding area is filled with black circles on a grey background. For simplicity, let us call this image π.

In the second smallest picture (Figure 2.2b), the sub-squares in the bottom left and top right consist of π as in Figure 2.2a. The sub-squares in the top left and bottom right are filled with a copy of Figure 2.2a.

In the third smallest picture (Figure 2.2c), the sub-squares in the bottom left and top right consist of π as in Figure 2.2a. The sub-squares in the top left and bottom right are filled with a copy of Figure 2.2b. This pattern continues from one picture in the sequence to the next.



(a) (b)

(c) (d)

Figure 2.2: Some pictures generated by G_{lights}.

This gallery is generated by RCPG $G_{\text{lights}} = (V_{\text{N}}, V_{\text{T}}, P, (S, \sigma))$ where $V_{\text{N}} = \{S, A, B, C, F\}$, $V_{\text{T}} = \{w, b\}$ and P is the set of rules in Figure 2.3.

$$
\begin{aligned}
S &\rightarrow [B, A, A, B]\,(\emptyset; \{C\}) && (2.1)\\
B &\rightarrow [b, b, b, b, w, b, b, b, b]\\
A &\rightarrow C\,(\emptyset; \{S, F\}) \;\mid\\
&\qquad F\,(\emptyset; \{S, C, F\}) \;\mid\\
&\qquad w\,(\{F\}\,; \{S\}) && (2.2)\\
C &\rightarrow S\,(\emptyset; \{A\})\\
F &\rightarrow w\,(\emptyset; \{A, S\})
\end{aligned}
$$

Figure 2.3: Rules for grammar G_{lights}

The terminals 'w' and 'b', represent 'white' and 'black' circles respectively. The random context regulates the application of the rules. For example, Rule 2.1, can only be applied if there is no C present in the pictorial form, while Rule 2.2 can only be applied if there is F and no S in the pictorial form.

Table 2.1 illustrates a derivation of Figure 2.2a. In the first column, we give the sequence of the pictorial forms in the derivation. In the second column, we provide the rule that is applied to the pictorial form to generate the pictorial form in the next row.

Table 2.1: Derivation of Figure 2.2a

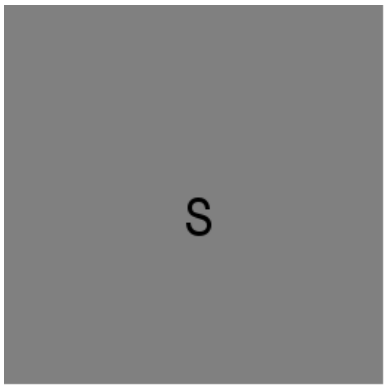
Pictorial Form	**Next rule**
	$S \rightarrow [B, A, A, B]\,(\emptyset; \{C\})$

Table 2.1 continued from previous page

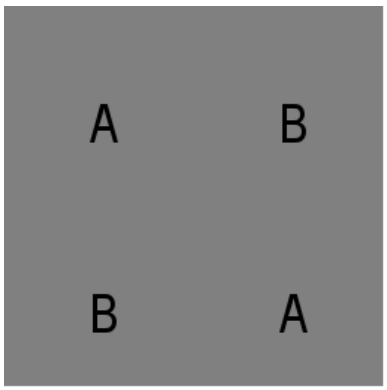
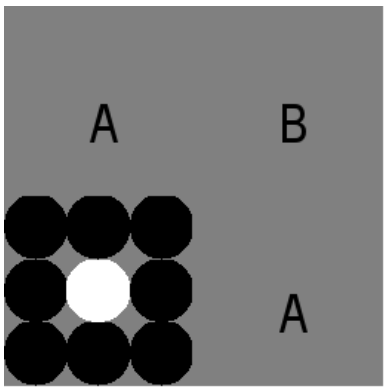
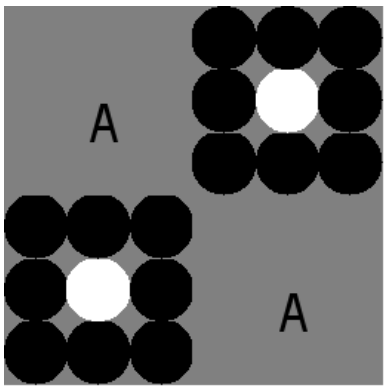
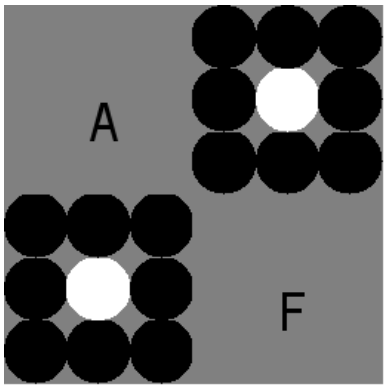
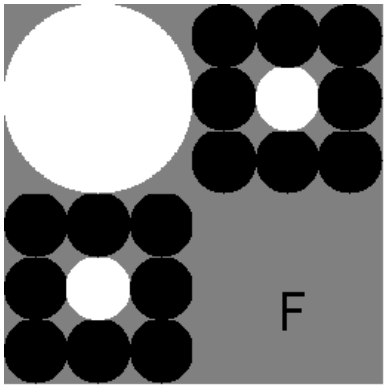
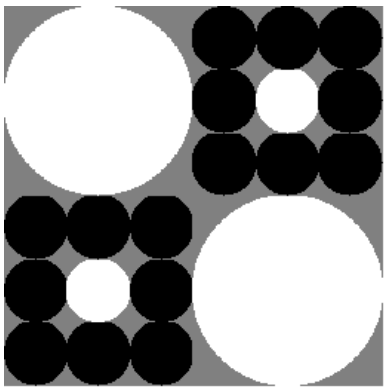
Pictorial Form	Next rule
	$B \rightarrow [b, b, b, b, w, b, b, b, b]$
	$B \rightarrow [b, b, b, b, w, b, b, b, b]$
	$A \rightarrow F\left(\emptyset; \{S, C, F\}\right)$
	$A \rightarrow w\left(\{F\}; \{S\}\right)$

Table 2.1 continued from previous page

Pictorial Form	Next rule
	$F \rightarrow w\,(\emptyset; \{A, S\})$
	

2.1.3 Generalised Random Context Picture Grammars

Generalized random context picture grammars (GRCPGs) create pictures using terminals which are subset of the Euclidean plane, by applying functions created by production rules to terminals [23, 24, 26, 50]. GRCPGs are context free grammars with regulated rewriting. They use context to enable or inhibit the application of production rules.

The following definitions were taken from [26, 50].

Definition 2.5. Let S be any set. Then $\mathfrak{P}(S)$ denotes the power set of S.

Definition 2.6. In [26] GRCPG are defined as tuple $G = (V_N, V_T, V_F, P, (S, \epsilon))$ has a finite alphabet V of labels, consisting of disjoint subsets V_N of variables, V_T of terminals and V_F of function identifiers. P is a finite set of productions of the form $A \rightarrow (A_1, f_1)(A_2, f_2)\ldots(A_t, f_t)(\mathcal{P}, \mathcal{F})$, where $A \in V_N, A_1, \ldots, A_t \in V_N \cup V_T, f_1, \ldots, f_t \in V_F$ and $\mathcal{P}, \mathcal{F} \subseteq V_N$. Finally, there is an initial configuration (S, ϵ), where $S \in V_N$ and ϵ denotes the empty string.

Definition 2.7. A pictorial form Π is a finite set $\{(A_1, \varphi_1), (A_2, \varphi_2), \ldots, (A_t, \varphi_t)\}$, where $A_1, \ldots, A_t \in V_N \cup V_T$ and $\varphi_1, \ldots, \varphi_t \in V_F^*$. The set $\{A_1, \ldots, A_t\}$ is denoted by $l(\Pi)$.

Definition 2.8. For an GRCPG G and pictorial forms Π and Γ, $\Pi \Longrightarrow_G \Gamma$ is written if there is a production $A \to (A_1, f_1)(A_2, f_2) \ldots (A_t, f_t)(\mathcal{P}; \mathcal{F})$ in G, Π contains an element $(A, \varphi), l(\Pi \backslash \{(A, \varphi)\}) \supseteq \mathcal{P}$ and $l(\Pi \backslash \{(A, \varphi)\}) \cap \mathcal{F} = \emptyset$, and $\Gamma = (\Pi \backslash \{(A, \varphi)\}) \cup \{(A_1, \varphi f_1), (A_2, \varphi f_2), \ldots, (A_t, \varphi f_t)\}$. As usual, $\Longrightarrow_G^*$ denotes the reflexive transitive closure of $\Longrightarrow_G$.

Definition 2.9. A picture is a pictorial form Π with $l(\Pi) \subseteq V_T$.

Definition 2.10. The gallery $\mathcal{G}(G)$ generated by a GRCPG G is the set of pictures Π such that $\{(S, \epsilon)\} \Longrightarrow_G^* \Pi$.

Definition 2.11. The gallery of a GRCPG G is rendered by specifying functions $\Psi_G : V_T \to \mathfrak{P}(\mathbb{R})^2$ and $\Upsilon_G : V_F \to F(\mathbb{R})^2$, where $F(\mathbb{R})^2 = \{g \mid g : \mathbb{R}^2 \to \mathbb{R}^2$. This yields a representation of a picture $\Pi = \{(A_1, \varphi_1), (A_2, \varphi_2), \ldots, (A_t, \varphi_t)\}$ in $\mathbb{R}^2$ by

$$r(\Pi) = \bigcup_{i=1}^{t} \Upsilon_G(\varphi_i)(\Psi_G(A_i)),$$

where Υ_G has been extended to V_F* in the obvious manner, $\Upsilon_G(\epsilon)$ representing the identity function id.

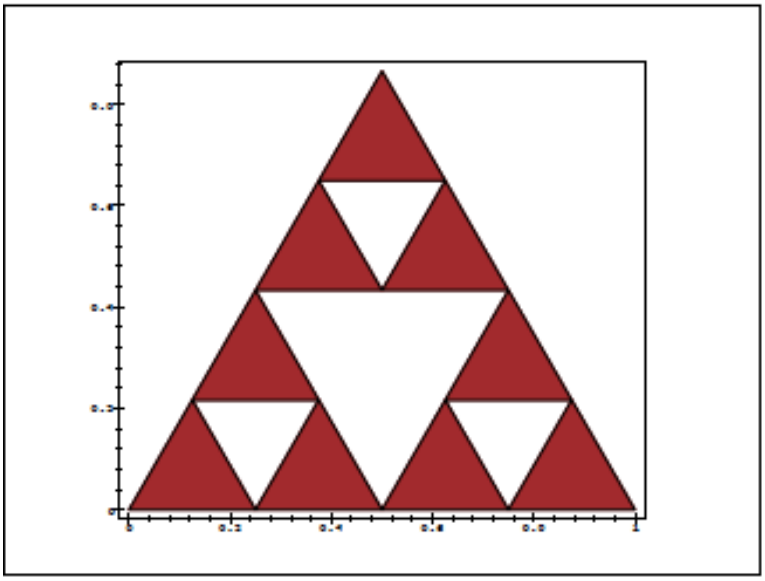


Figure 2.4: $\Psi_G(\{b\})$ is a dark triangle

The following example was taken from [26] and illustrates the GRCPGs concepts.

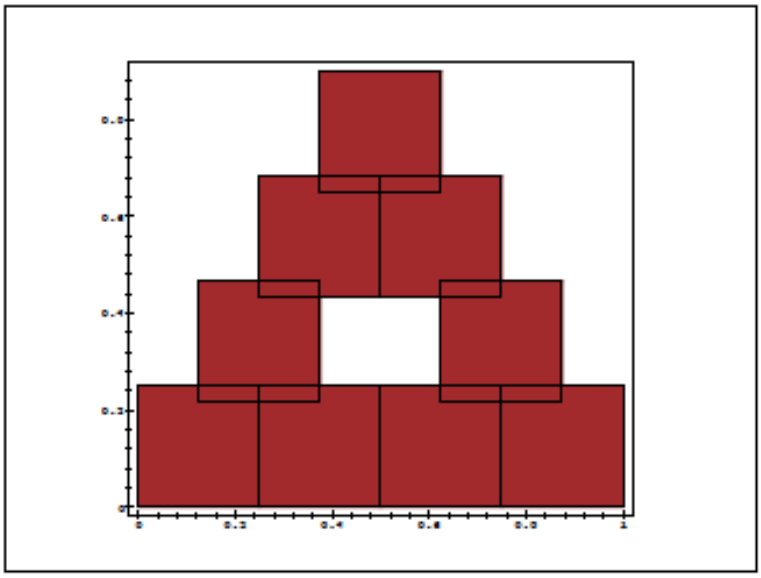


Figure 2.5: $\Psi_G(\{b\})$ is a dark square

$$
\begin{aligned}
S &\to \{(T; g_{lb})(T; g_{rb})(T; g_t)\}\,(\emptyset; \{U\}) \\
T &\to U\,(\emptyset; \{S, F\}) \mid \\
&\quad\; F\,(\emptyset; \{S, U, F\}) \mid \\
&\quad\; b\,(\emptyset; \{F\}) \\
U &\to S\,(\emptyset; \{T\}) \\
F &\to b\,(\emptyset; \{T\})
\end{aligned}
\tag{2.3}
$$

Figure 2.6: Rules for grammar G_{gasket}

Example 2.2. The following grammar generates the typical iteration sequence of the Sierpiński gasket, using the GRCPG $G_{\text{gasket}} = (\{S, T, U, F\}, \{b\}, \{g_{lb}, g_{rb}, g_t\}, P, (S, \epsilon))$ where P is the set in Figure 2.6.

Example of the pictures are Figure 2.4 and Figure 2.5.

2.1.4 Bag Context Tree Grammars

A bag context tree grammar (BCTG) is a regular tree grammar with regulated rewriting [18]. In BCTG, the application of a rule is regulated by a k-tuple of integers, the so-called bag, which changes during a derivation. A rule in the grammar can only be applied if the bag at that point is within the scope defined by the lower and upper limits of the rule. When a rule is applied, it causes the bag values to change by adding the bag adjustment,

which is part of the rule. For tree grammars, it was shown that bag context is strictly more powerful than random context [18, Theorem 5.1].

Definition 2.12. As defined in [17], a regular tree grammar is a quadruple $G = (N, \Sigma, P, S)$ consisting of

- a set N of variables of rank 0,
- a set Σ of terminals, with $N \cap \Sigma = \emptyset$,
- a finite set $P \subseteq N \times T_{\Sigma \cup N}$ of rule of the form $A \to r$, where $A \in N$ and $r \in T_{\Sigma \cup N}$; and
- an initial variable $S \in N$.

A term $t \in T_{\Sigma \cup N}$ directly derives $t' \in T_{\Sigma \cup N}$ and is written as $t \Rightarrow_R t'$, if there is a production $A \to s$ in P such that t' is obtained from t by replacing the occurrence of A in t with s. The generated language by G is $L(G) = \{t \in T_\Sigma \mid S \overset{*}{\Rightarrow}_R t\}$, where $\overset{*}{\Rightarrow}_R$ denotes the reflexive and transitive closure of $\Rightarrow_R$.

Definition 2.13. As defined in [18], a bag context (BC for short) tree grammar is a sextuple $G = (N, \Sigma, P, S, I, \beta_0)$ consisting of

- a finite alphabet N of variables of rank 0;
- a finite alphabet Σ of terminals with $N \cap \Sigma = \emptyset$;
- a finite set P of rules of derivation;
- an initial variable $S \in N$;
- a finite bag index set, I; and
- a vector $\beta_0 \in \mathbb{Z}^I$, the initial bag.

A rule in P has the generic form $A \to t(\,(\lambda, \mu; \delta))$, where $A \in N, t \in T_{\Sigma \cup N}, \lambda, \mu \in {\mathbb{Z}_\infty}^I$ are the lower and upper limits respectively, and $\delta \in \mathbb{Z}$ is the bag adjustment.

The following example adapted from Drewes et al. [18] demonstrates how BCTGs work.

Example 2.3. BCTG $G_{\text{tree}} = (N, \Sigma, R, (S, \sigma), [1, 2, 3]\,, (1, 0, 0))$, where $N = \{S, A, B\}$, $\Sigma = \{f^{(2)}, a^{(0)}\}$, and R is the set of rules in Figure 2.7.

The bag in G_{tree} has three positions which count the number of occurrences of S, A and B respectively, in the developing tree. We show an example of a tree generated by G_{tree} in Figure 2.8. Table 2.2 illustrates the evolution of the bag in a derivation of Figure 2.8. We illustrate the

$$S \to f[A, A]\left((1,0,0),(\infty,\infty,0);(-1,2,0)\right) \tag{2.4}$$
$$A \to B\left((0,1,0),(0,\infty,\infty);(0,-1,1)\right) \tag{2.5}$$
$$B \to S\left((0,0,1),(\infty,0,\infty);(1,0,-1)\right) \tag{2.6}$$
$$S \to a\left((1,-\infty,-\infty),(\infty,0,0);(-1,-1,-1)\right) \tag{2.7}$$

Figure 2.7: Rules for grammar G_{tree}

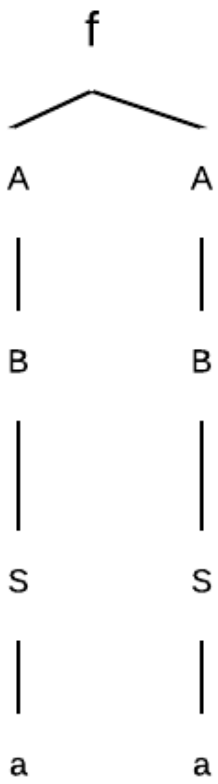


Figure 2.8: An example of a tree generated by G_{tree}

evolution of the bag during this derivation in the first column. In the second column, we provide the rule that is applied to the pictorial form to generate the pictorial form in the next row.

The bag context regulates the application of the rules. For example, Rule 2.4 and Rule 2.7 are the only rules applicable at the beginning of the cycle, while repeated application of Rule 2.7 terminates the cycle as they cause the second and third position to be negative.

Table 2.2: The evolution of the bag in a derivation of Figure 2.8

Current Bag	**Next rule**			
$(1,0,0)$	$S \to f[A,A]$	$((1,0,0),$	$(\infty,\infty,0);$	$(-1,2,0))$
$(0,2,0)$	$A \to B$	$((0,1,0),$	$(0,\infty,\infty);$	$(0,-1,1))$
$(0,1,1)$	$A \to B$	$((0,1,0),$	$(0,\infty,\infty);$	$(0,-1,1))$
$(0,0,2)$	$B \to S$	$((0,0,1),$	$(\infty,0,\infty);$	$(1,0,-1,))$
$(1,0,1)$	$B \to S$	$((0,0,1),$	$(\infty,0,\infty);$	$(1,0,-1,))$
$(2,0,0)$	$S \to a$	$((1,-\infty,-\infty),$	$(\infty,0,0);$	$(-1,-1,-1,))$
$(1,-1,-1)$	$S \to a$	$((1,-\infty,-\infty),$	$(\infty,0,0);$	$(-1,-1,-1,))$
$(0,-2,-2)$				

2.1.5 Lindenmayer Systems

Lindenmayer systems (L-systems) were developed by Lindenmayer as a theoretical framework for the development of simple multicellular organisms [54]. L-systems are traditionally used in the simulation and modelling of plants [57, 71, 72]. Recently, L-systems have been used in diverse applications such as the generation of fractals, landscape design, research and education in botany and ecology, music rendering, animation and decision support in agriculture [9, 23, 57, 70, 72, 73, 75, 96].

L-systems come in both context-sensitive and context-free forms. There are many classes of L-systems, for example, DOL-systems, stochastic L-systems and ETOL-systems. DOL-systems is the simplest class of L-systems which is deterministic and context free [69]. A DOL-system generates identical structures even with repetitive usage. This was a motivation for development of stochastic L-systems. In stochastic L-systems, probability is used in each production. This class of L-systems is capable of generating variations in structures using the same set of production rules [73]. Stochastic L-systems ability to generate variations in structures is very important in the generation of structurally similar pictures, as the pictures produced have similar properties but are not identical. Figure 2.9 is an example of a set of pictures which share similar properties produced by a stochastic L-systems

[73]. The problem with the pictures in Figure 2.9 is that there are pictures very different from others, such as the second tree. In ETOL-system, E stands for extended, T for tables, 0 (zero) means the rules do not depend on any context, and L refers to Lindenmeyer [10, 16]. The use of tables in ETOL-systems gives it the capability to describe some natural and attractive structures otherwise difficult to define [16]. Figure 2.10 is an example of an ETOL-system which generates a sequence of pictures using one table. An example of an ETOL-system that uses more than one table to create images with different variations which have similar properties is shown in Figure 2.11 [16, pp.82].

The ability of generating variations in images is very important in the generation of similar pictures, as the pictures are supposed to look similar but not identical. L-Systems also have the capability to generate images with variations and can, thus, generate structurally similar pictures.

The following definitions of L-systems were adapted from Prusinkiewicz [69]. Let V denote an alphabet, V^* is the set of all words over V and V^+ is the set of all non-empty words over V.

Definition 2.14. A OL-system is an ordered triplet $G = (V, w, P)$ consisting of

- an alphabet V
- an axiom $w \in V^+$ and
- a finite set of rules $P \subset V \times V^*$.

A rule in P has the form $a \to X$ for a pair (a, X). We call a the predecessor and X the successor of this production. For any $a \in V$, there is at least one word $X \in V^*$ such that $a \to X$. A OL-system is deterministic if and only if for each $a \in V$ there is exactly one $X \in V^*$ such that $a \to X$.

The following is an interpretation of a DOL-system in terms of a TOL-system.

Definition 2.15. A TOL-system is an ordered quadruplet $G = (V, w, P, T)$. The alphabet V, the axiom w and the set of rules P are as in the definition of an OL-system and T is a finite non-empty collection of subsets of P, called tables. It is assumed that for each table $t \in T$ and each letter $a \to V$ there is at least one production $p \in t$ with the predecessor a.

Definition 2.16. A stochastic OL-system is an ordered quadruplet $G = (V, w, P, \pi)$. The alphabet V, the axiom w and the set of rules P are as in the definition of a OL-system. The probability function is denoted by $\pi : p \to R^+$, and maps the set of rules into the set of positive real numbers. The values of π are called the probability factors.

Definition 2.17. The subset of all rules of P which have the predecessor a is denoted by $P'(a) \subset P$.

We will call the derivation $\mu \Rightarrow v$ a stochastic derivation in G_π if for each occurrence of the letter a in the word μ the probability of selecting rule $p_i \in P'(a)$ is defined as:

$$prob(p_i) = \frac{\pi(p_i)}{\sum_{p_k=P'(a)} \pi(p_k)} \tag{2.8}$$

According to the formula 2.8, in one derivation step, different rules with the same predecessor can be applied to various occurrences of the same letter.

The following is a simple example of stochastic L-system adapted from Prusinkiewicz [69].

Example 2.4. Figure 2.12 provides a set of rules in the generation of the gallery in Figure 2.9. The probability factors are listed after the rules. According to the formula 2.8, all the rules have the same probability of being selected:

$$prob(p_1) = prob(p_2) = prob(p_3) = \frac{0.3}{0.3 + 0.3 + 0.3} = 0.33 \tag{2.9}$$

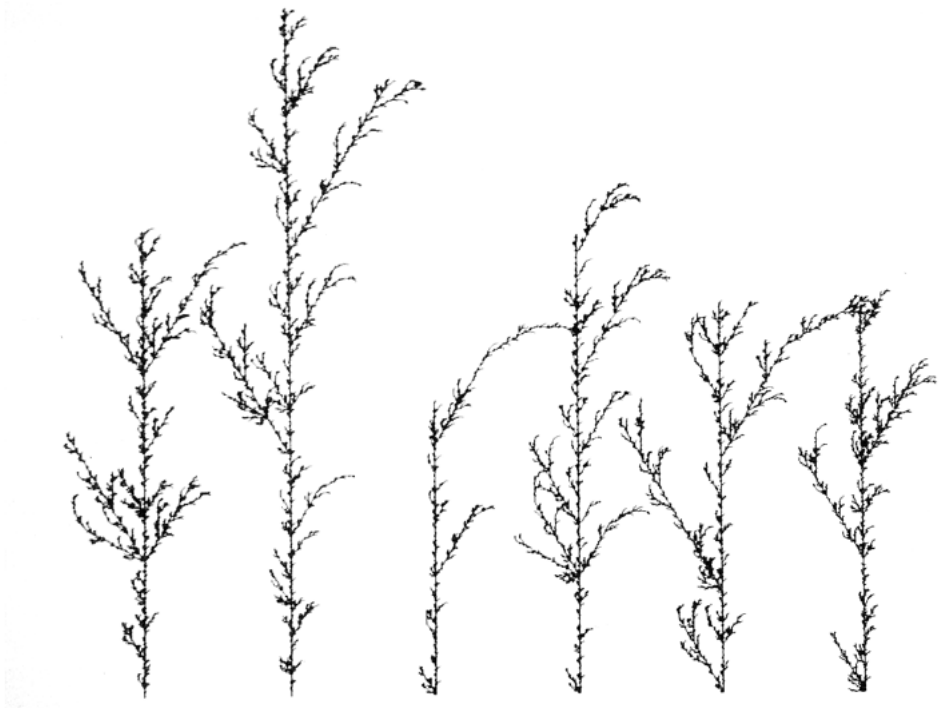


Figure 2.9: Some pictures generated by Stochastic L-systems [69]

The following example was adapted from Drewes [16]

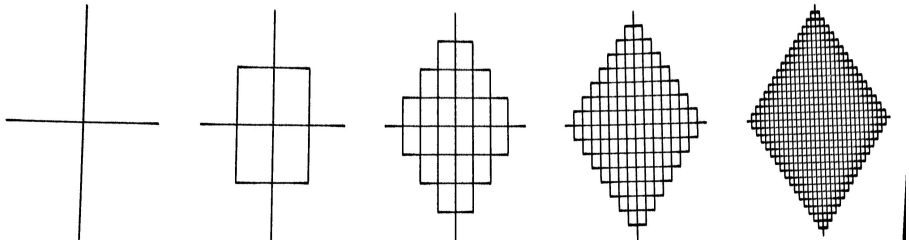


Figure 2.10: ETOL picture sequence [16]

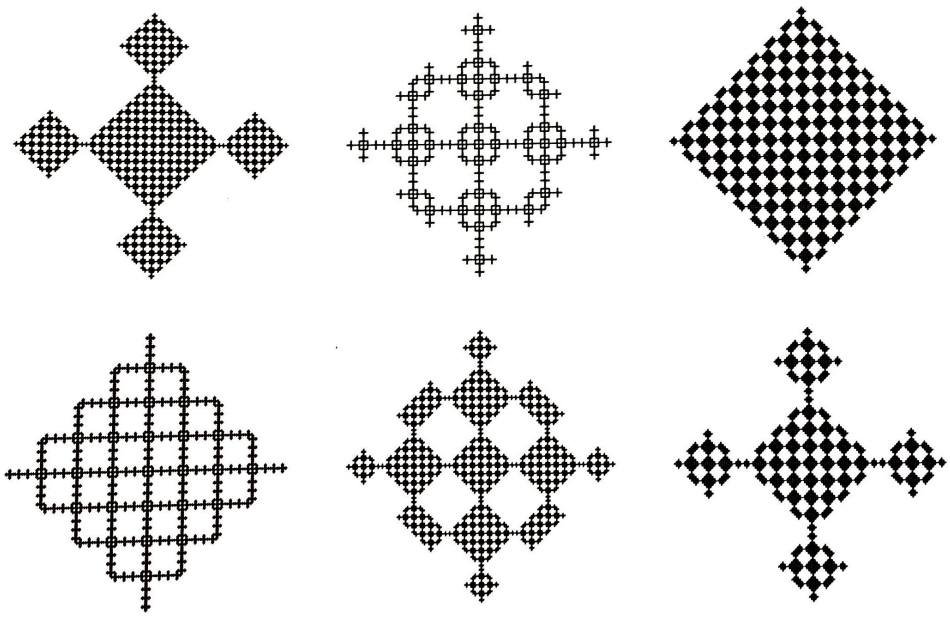


Figure 2.11: Variety of pictures by an ETOL chain-code picture language [16]

w	F	
p_1	$F \to F[{}'{+}F]F[{}'{-}F]F$	0.3
p_2	$F \to F[{}'{+}F]F$	0.3
p_3	$F \to F[{}'{-}F]F$	0.3

Figure 2.12: Stochastic L-system rules

$$l \to ldul$$
$$r \to rudr$$
$$u \to ulru$$
$$d \to drld$$

Figure 2.13: The first ETOL-system table

Example 2.5. Consider the table in Figure 2.13

This table on its own generates a picture sequence shown in Figure 2.10, from the axiom $udrldulr$. We now add the table in Figure 2.14. This table is obtained from the first by removing u and d from right-hand sides of the rules l and r and vice versa. A greater variety of pictures can be obtained

$$l \to ll$$
$$r \to rr$$
$$u \to uu$$
$$d \to dd$$

Figure 2.14: The second ETOL-system table

by addition of the table in Figure 2.14. Some of the pictures are shown in Figure 2.11. The rules in this example are deterministic, and the only non-determinism lies in the selection of the tables. The symbols l, r, u and d represents left, right, up and down respectively.

2.1.6 Array Grammars

Array grammar definitions were motivated by the kolam patterns [77, 78]. Kolam are traditional art drawings which are used to decorate the front of houses found in southern Indian villages [78]. There are two different types of array grammars. These are isometric grammars and non-isometric grammars [86]. Isometric grammars preserve the shape of the rewritten sub-array after rule application, this is because both the left and right side of rule are geometrically identical. Whereas in non-isometric grammars the application of a rule may change the dimension of the rewritten sub-array [86]. The rewriting rules in array grammars are either regular, context free or context sensitive [77]. Derivations are determined by the state of row and column sequence [77]. Inspired by different applications in relation to image analysis and processing, several kinds of array grammars have been proposed [31, 32, 59, 81, 85, 86, 94]. There are array grammars that use permitting features explored in [81, 86]. Subramanian et al. [86] found that it was simpler associating permitting symbols with rules than the usual pattern matching. Moreover, the use of permitting symbols reduced the number of components needed to produce a set of picture arrays [81, 86].

We now provide the definition of isometric array grammars from Subramanian et al. [85, 86].

Definition 2.18. An array grammar $G = (V_{\mathrm{N}}, V_{\mathrm{T}}, P, S, \#)$, has a finite alphabet V of labels, consisting of disjoint subsets V_{N} of variables and V_{T}

of terminals, $S \in V$ is the start symbol and # is the blank symbol. P is a finite set of array production rules of the form $r : \alpha \to \beta$, where α and β are arrays over $V \cup \#$ satisfying the following conditions:

1. the arrays α and β have geometrically identical shapes,
2. α contains at least one element of V_{N},
3. the terminals in α are not erased by the application of a rule, and
4. the connectivity of the rewritten array is preserved on the application of the rule.

The following is an example of array grammar adapted from [85].

Example 2.6. Consider a context free grammar G_{array} with rules in Figure 2.16, where $V_{\mathrm{N}} = \{S, A, B, C\}$ and $V_{\mathrm{T}} = \{a\}$. G_{array} generates an array with three arms over $\{a\}$, where the length equals to the number of a's in an arm, the arms' lengths might vary. Rule 2.10 is applicable only at the beginning of derivation, in a derivation starting with S. Application of Rule 2.11 can be repeated as many times as needed to increase the upper arm's length, and the application of Rule 2.14 terminates its growth. Similarly, application of Rules 2.12 and 2.13 increases the length of horizontal left and right arms which can be terminated by the application of Rules 2.15 and 2.16. Figure 2.16 is an example of a picture generated by G_{array} representing the mathematics symbol for perpendicularity.

$$\begin{array}{ccc} & \# & \\ \# & S & \# \end{array} \to \begin{array}{ccc} & & A \\ C & a & B \end{array}, \tag{2.10}$$

$$\begin{array}{c} \# \\ A \end{array} \to \begin{array}{c} A \\ a \end{array}, \tag{2.11}$$

$$B\# \to aB \tag{2.12}$$

$$\#C \to Ca \tag{2.13}$$

$$A \to a \tag{2.14}$$

$$B \to a \tag{2.15}$$

$$C \to a \tag{2.16}$$

Figure 2.15: Production rules of G_{array}

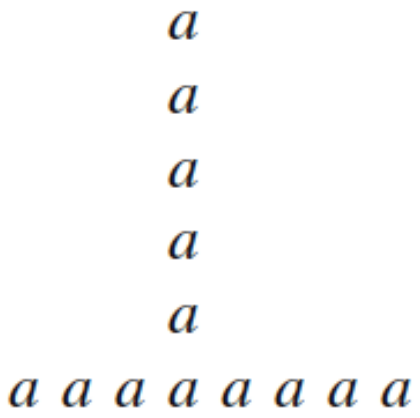


Figure 2.16: A picture generated by G_{array}

2.1.7 Puzzle Grammars

Puzzle grammars were introduced by Nivat et al. [61] with the desire to solve tiling problems [61, 82–84]. Basic puzzle grammars (BPGs) which are a subclass of puzzle grammars generalise regular array grammars (RAGs) [82]. BPGs can generate a sequence of an isosceles right-angled triangles [84] while RAGs can not [79]. Both BPGs and RAGs are deterministic in that they generate pictures that are a sequence [79]. In order to bring non-determinism in pictures a class of puzzle grammars called stochastic puzzle grammars was introduced in Siromoney et al. [79]. The benefit of stochastic grammars is the reduced time complexity as a result of productions being selected about their probabilities rather than random selection [79].

The following definition was adapted from Subramanian et al. [82, 83].

Definition 2.19. A basic puzzle grammar $G = (V_{\text{N}}, V_{\text{T}}, P, S)$ has a finite alphabet V of labels, consisting of disjoint subsets V_{N} of variables and V_{T} of terminals. P is a finite set of productions of the form in Figure 2.17, where $A, B \in N$ and $a \in T$.

Every derivation begins with S written in a unit cell in the two-dimensional plane, with all the other cells containing the blank symbol # not in $V_{\text{N}} \cup V_{\text{T}}$. In a derivation step, denoted $\Rightarrow$, a variable A in a cell is replaced by the right-hand member of a rule whose left-hand side A. In this replacement, the circled symbol of the right-hand side of the rule used, occupies the cell of the replaced symbol and the non-circled symbol of the right-hand side occupies the cell to the right or left or above or below the cell of the replaced symbol, depending on the type of rule used. The replacement is possible only if the cell to be filled in by the non-circled symbol contains a #.

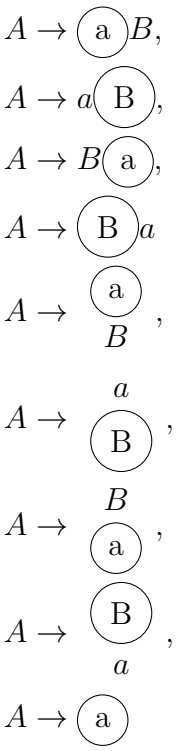


Figure 2.17: A basic puzzle grammar production rules

The set of pictures generated by G, denoted by $l(G)$ is the set of connected, digitized finite arrays over V_{T}, which can be derived in one or more step from the start symbol.

Definition 2.20. A puzzle grammar is called non-overlapping, if it satisfies the following condition: Whenever a variable A is to be rewritten in a derivation step by a rule $A \to a$ in G, with a as the circled symbol in a, then all the cells to be filled in by the non-circled symbols of a, should contain only the blank symbol; otherwise, the puzzle grammar is said to be overlapping [83].

The following example was adapted from Subramanian et al. [83].

Example 2.7. Consider Figure 2.18. Figure 2.18 is generated by a non-overlapping BPG G_{puzzle}=$(V_{\mathrm{N}}, V_{\mathrm{T}}, P, S)$ where $V_{\mathrm{N}} = \{\ S, A, B, B_1, B_2, C, C_1, C_2\}$, V_{T}=a and P is a set of rules in Figure 2.19.

We now give an example of solid isosceles right triangles adapted from Siromoney et al. [79].

Example 2.8. Solid isosceles right triangles $[T'_n : n \geq 2]$. Let $G_{\mathrm{triangles}} = (\{A, B\}, \{x\}, J, I)$ where

$$I = \begin{matrix} x \\ AB \end{matrix} \tag{2.17}$$

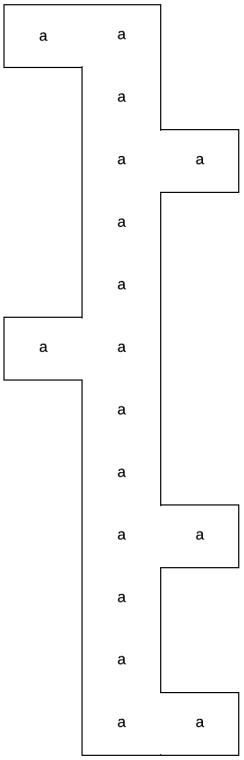


Figure 2.18: A picture generated by G_{puzzle}

and the tables in J are

$$T_1 = [A \to \underset{AB}{\text{(x)}} \quad B \to \underset{B}{\text{(x)}}\ ]$$
$$T_2 = [A \to \text{(x)} \quad B \to \text{(x)}]$$

Clearly, $T'_n (n \geq 2)$ is generated by applying the axiom, followed by $T_1(n-2)$ times and T_2 once. Some pictures generated by $G_{\text{triangles}}$ are shown in Figure 2.20.

2.1.8 Iterated Function Systems

Iterated function systems (IFSs) were introduced by Barnsley and co-workers as a means of generating fractals and compressing images [9, 23, 30]. A fractal is a result of the union of several transformed copies of itself. IFSs are usually contractive; they make smaller shapes by bringing points closer together [14].

IFSs are commonly used in the study of fractals and are, perhaps, one of the best means of mathematical picture generation [50]. IFSs do not use

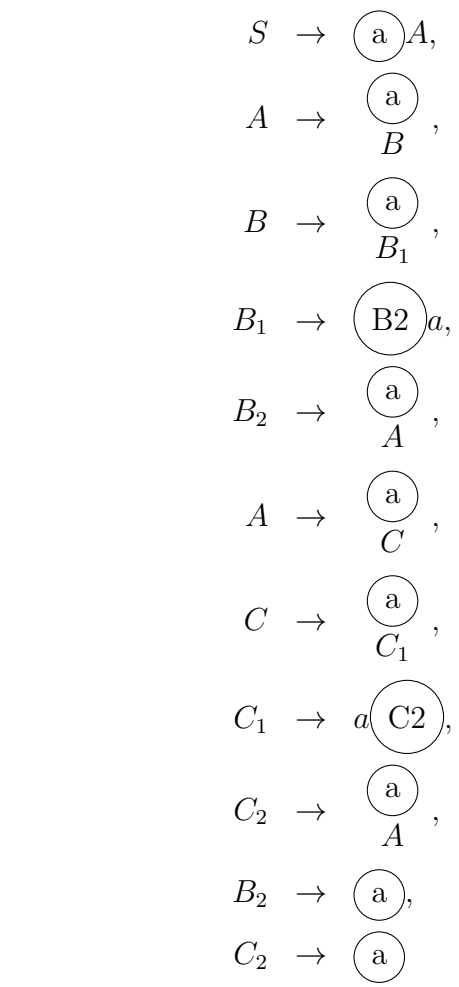


Figure 2.19: Production rules for the grammar G_{puzzle}

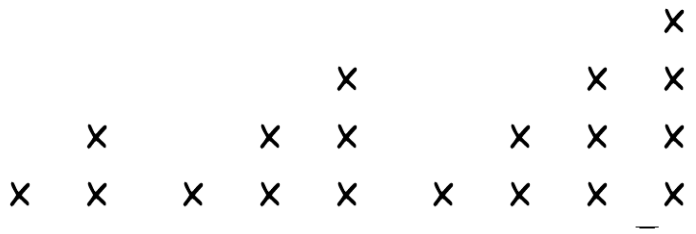


Figure 2.20: Some pictures generated by $G_{\text{triangles}}$

context in picture production. IFSs tend to generate pictures with a high degree of self-similarity which results in a limited range of pictures produced. There is a generalised type of IFSs that is considered more powerful than IFSs known as mutually recursive function systems (MRFSs) [8, 14, 50]. Their development was motivated by the desire to explore wider ranges of fractal-like images that do not display such high degrees of self-similarity [50] and can shrink the image details enormously [8]. There are several translations between IFSs and MRFSs and other forms of syntactic picture generation [9, 14, 30, 50]. In translating MRFSs to other picture grammars, it should be noted that a picture generated by MRFSs have the same level of refinement in all regions [9, 50]. It was shown that uniformly growing D0L-Systems can be translated into MRFSs [9]. In Ewert and van der Walt [30], they have shown that IFSs can be translated into equivalent GRCPGs. GRCPGs that use only forbidding context can create all galleries that are produced by IFSs [30]. Forbidding context was used to ensure that the pictures generated have uniform refinement by restricting the application of production rules. Kruger and Ewert [50] have presented the translation of MRFSs to GRCPGs which is an extension of the later. GRCPGs were shown to be more powerful than IFSs due to their use of context in the application of rules [30]. An example of a set of pictures which are similar generated by MRFSs is shown in Figure 2.21. Although the pictures in Figure 2.21 are similar, it is difficult to tell the difference between some of the pictures, for example, the last two pictures on the bottom right. Moreover, in some of the pictures, it is too easy to tell the difference.

The following pictures were taken from [14]

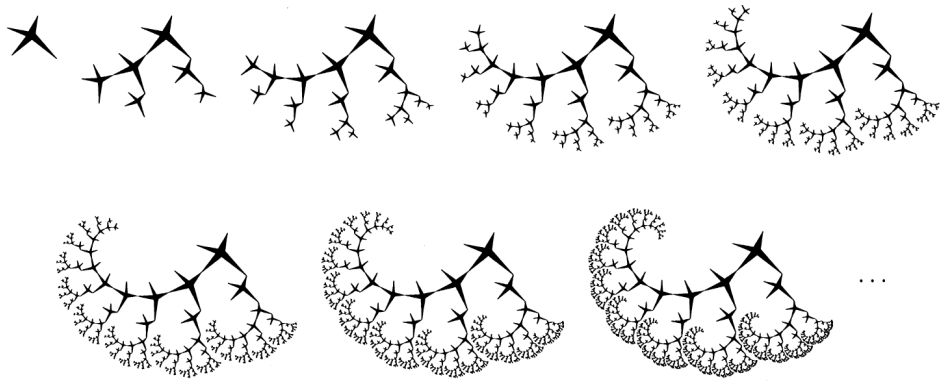


Figure 2.21: Example of pictures produced by MRFS

2.1.9 Collage Grammars

Collage grammars are picture grammars based on hyperedge replacement which originates from the area of graph transformation [19, 49]. A collage

is a picture which consists of sets of parts, where each part is a geometric object such as circles, rectangles and triangles [16, 49]. In collage grammars, collages are derived from collages. Thus they do not require extra pictorial interpretation [19]. Collage grammars come in various classes such as context-free collage grammars and ETOL collage grammars [16, 49]. An ETOL collage grammar uses sets of tables which allows simultaneous replacement of hyperedges. ETOL collage grammars are more powerful than the context-free collage grammars [16].

The following definition was adapted from Drewes et al. [19].

Definition 2.21. A context-free collage grammar is a regular tree grammar $G = (V_{\mathrm{N}}, \Sigma, P, S)$ where V_{N} is a finite set of variable, Σ is a collage signature P is a set of productions. The language generated by G is a context-free language.

Below is an example of a collage grammar adapted from Drewes [16].

Example 2.9. Consider the grammar G_{collage} which is linear and uses only a single variable. Figure 2.22 shows the rules. Figure 2.23 provides the initial steps of derivation in G_{collage} and Figure 2.24 are some pictures generated by G_{collage}.

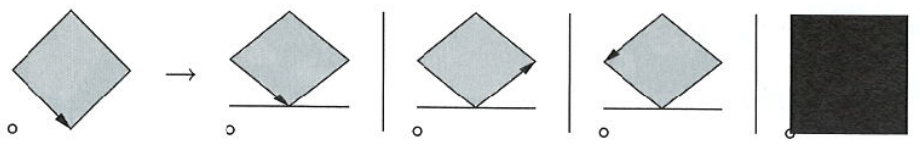


Figure 2.22: Production rules of G_{collage}

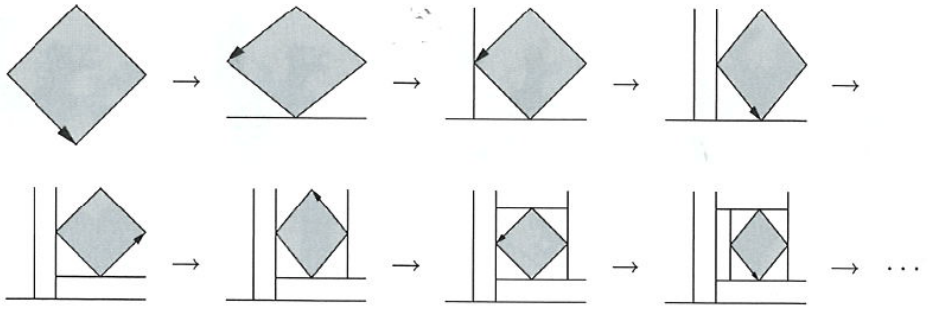


Figure 2.23: A derivation in G_{collage}

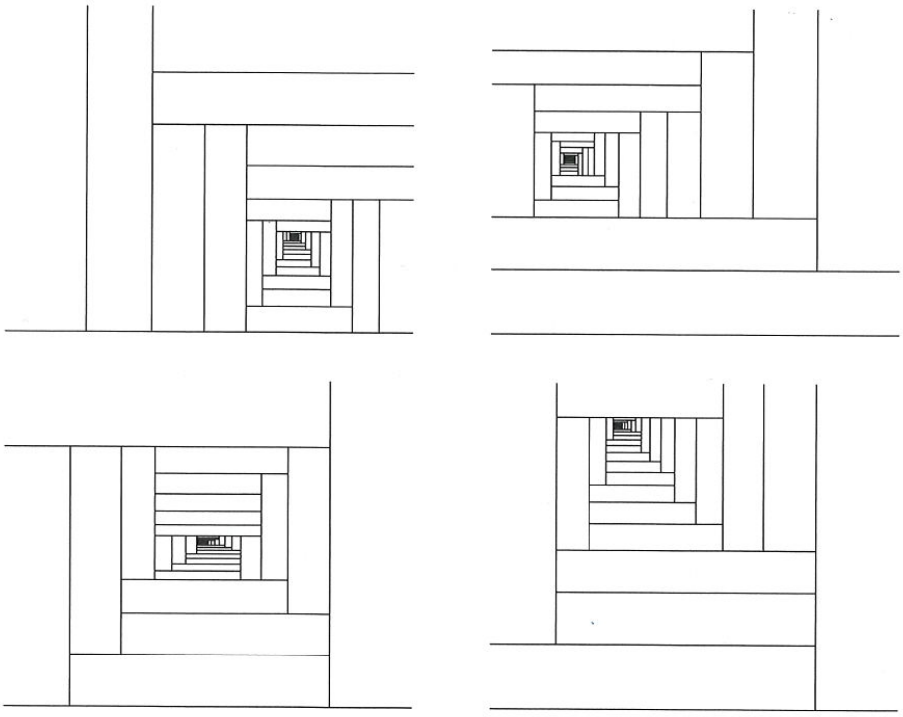


Figure 2.24: Some pictures generated by G_{collage}

2.1.10 Shape Grammars

The shape grammar was introduced by Stiny and Gips [80]. In shape grammars, definitions are made with regards to shapes. Their production rules are shapes rather than symbols [42, 80]. There are many applications of shape grammars in literature such as generation of paintings and sculptures and architecture designs [16, 42, 76, 80]. Shape grammars consider context-free and ETOL collage grammars as its natural special case [16].

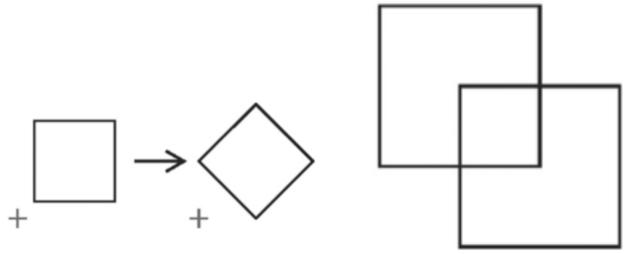
The following definition is adapted from [76, 80].

Definition 2.22. A shape grammar is a quadruple $G = (V_{\text{T}}, V_{\text{M}}, P, S)$ consisting of

- a finite set of shapes V_{T},
- a finite set of shapes V_{M} such that $V_{\text{T}} * \cap V_{\text{M}} = \emptyset$,
- a finite set of production rules P of the form $u \to v$,
- an initial shape S.

In shape grammars, a shape is generated by starting with an initial shape S and recursively applying the production rules. Another shape is generated as a result of rules application. There are challenges when it comes to

implementations of shape grammars such as specification of part structures, which can change dynamically as the shape evolves, limited capability of shape grammars due to the adoption of finite fixed structures [46].



(a) (b)

Figure 2.25: Example of a shape grammar

The following is an example of shape grammar adapted from Jowers et al. [46].

Example 2.10. An example of a simple shape grammar is shown in Figure 2.25. There is only one shape rule, Figure 2.25a, and an initial shape is Figure 2.25b. For a rule to be applied in shape grammars, it must first recognise the shape in the left-hand side of the rule embedded as part of a shape. Then the recognised shape is replaced by the shape on the right-hand side of the rule. Figure 2.26, shows a network of shapes generated by repeated application of the rule. Despite the simplicity of this example, the computation of shape is complicated due to the need to recognise all squares in the left-hand side shape.

2.1.11 Summary of Picture Grammar Types

We have discussed several formal grammars in order to determine their strength and weakness. In this research, we produce in a controlled manner a finite number of structurally similar pictures. Structurally similar are pictures that share some characteristics based on how they were generated, for example, the pictures contain the same sub-pictures or they have been refined the same number of times. One crucial factor in the generation of structurally similar pictures is producing pictures with variation. Moreover, the simplicity of controlling the application of rules is also essential. The grammar types discussed have a variety of characteristics such as, there are the ones that generate pictures that form part of a sequence, those that generate pictures with variations, context-free grammars, context-sensitive

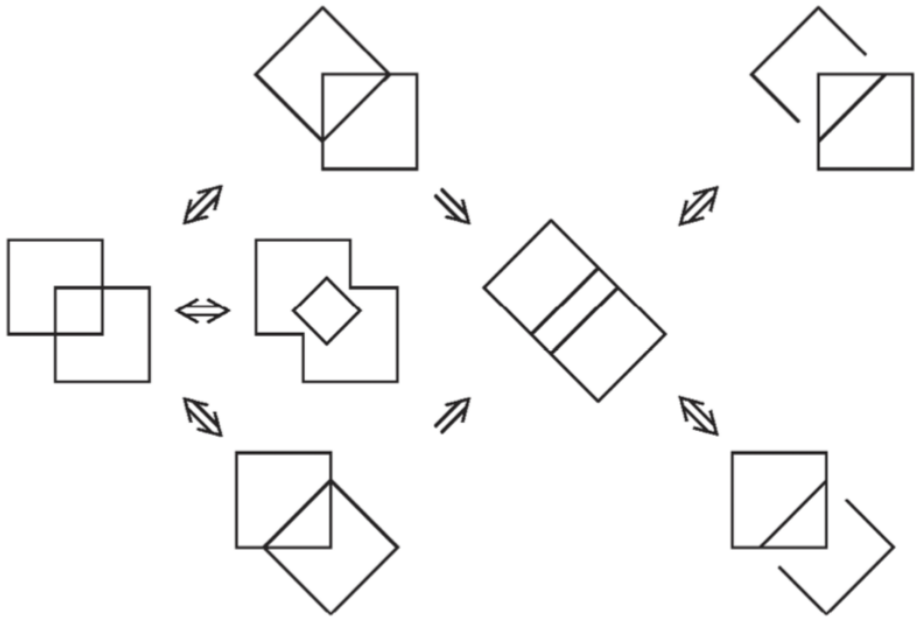


Figure 2.26: Application of the shape grammar in Figure 2.25

ones and which are context-free with regulated rewriting. While most grammar types consist of regular, context-free and context-sensitive classes, they differ on their strength.

We say pictures generated are sequential or form part of a sequence if, for all pictures produced at n level of refinement are identical. Example of the grammar types that generate sequential pictures is D0L-systems [69], isometric array grammars [86], basic puzzle grammars [79] and iterated function systems. These grammars can not generate variations on pictures even if the given grammar is used repeatedly due to its deterministic nature [69, 77]. If one needs to generate a variation on a picture, the grammar might need to be rewritten. The gallery of pictures generated by these grammars is merely a sequence. RCPGs can also generate sequential pictures such as Figure 2.2, although they can be programmed to generate pictures with variations. This is discussed in more detail in Chapters 3 and 4.

Many grammars are capable of generating pictures with variations such as RCPGs, stochastic L-systems and ET0L-systems, stochastic puzzle grammars. In Stochastic L-systems and stochastic puzzle grammars, it is possible to generate pictures with variations due to the use of probability in the application of rules. However, the use of probability in the application of rules makes it difficult to control their application. Similarly, ET0L-systems can generate variations in pictures due to its use of tables. ET0L-systems that uses more than one table can generate pictures which are not a sequence. Despite this, it is difficult to use tables to control the pictures that are

produced by a given picture grammar. Moreover, RCPGs are another picture grammar type that can generate pictures with variations due to its use of context. They are more powerful than IFS[30]. RCPGs are context-free grammars with regulated rewriting. Thus it is easier to control the application of rules with them. Hence by using a RCPG, it is easy to control the produced set of pictures.

We chose to use RCPGs in this research due to their strength over many grammar types in the framework of regulation of application of rules. Moreover, RCPGs were chosen to be used in the generation of structurally similar pictures because to the best of our knowledge, there are no defined methods for generating in a controlled way structurally similar pictures using RCPGs. In the tree and string case, BCTGs were found to be more powerful than RCPGs [18]. Thus, we define BCPGs in Chapter 3 based on RCPGs and BCTGs. There is a great potential in BCPGs in the generation of pictures due to regulated rewriting in the application of rules.

2.2 Mathematical Similarity Measures

Determining image similarity is a crucial element in many applications that require the comparison of images on different aspects, like colour, texture, layout and theme. Applications like search engines, image database retrieval systems, picture generators and visual password systems are some of the applications that may require determining the degree of similarity of images [34, 63]. Determining image similarity is important for our research as we generate similar pictures, it is, thus, necessary to evaluate if the produced images are indeed similar. Hence we examined different content based image retrieval (CBIR) systems to determine which mathematical similarity measures are best suited for our research.

2.2.1 Texture Descriptors

There are several CBIR systems based on texture features in the literature [56, 90]. These CBIR systems measure image similarity based on the texture of the image. Some of the texture features considered in determining the similarity are roughness, contrast, coarseness, line-likeness, directionality and regularity. Several researchers have studied the effectiveness of these CBIR systems based on texture features to determine the best similarity measure [55, 63]. The pictures generated in our study do not contain texture regions and, thus, CBIR systems based on texture are not suitable.

2.2.2 Colour Descriptors

There are many colour descriptors for measuring similarity in the literature. Colour histograms are the most common CBIR systems used, due to their effectiveness and ease of implementation [88]. However, the shortcoming of conventional colour histograms is their lack of spatial information which makes them intolerant of small changes in viewing positions [37, 38, 52, 55, 63]. Many CBIR systems that take into account spatial colour information have been proposed [6, 37, 38, 47, 52, 55, 87, 97]. One example of a CBIR system that takes spatial colour information into account is colour correlogram [37]. Colour correlogram has been shown to be more powerful than the traditional colour histograms [37, 38, 47, 52, 55, 97]. Despite its power compared with colour histograms, colour correlograms have severe computational complexity and tremendous memory requirement problems [47]. Another colour descriptor taking spatial colour information into consideration was introduced in Kiranyaz et al. [47]. This colour descriptor takes into account how humans perceive colours when extracting and representing the global and spatial colour properties [47]. Chatzichristofis et al. [6] introduced a new compact composite descriptor which combines colour and spatial colour distribution information, called spatial colour distribution descriptor. Okundaye et al. [63] found that this colour model provided better retrieval results for syntactically generated images than correlogram and the other earlier colour descriptors. For this reason, this colour descriptor was chosen to measure mathematical similarity in this study as the main feature of the images generated in this study is colour and the location of various coloured objects in the picture.

2.2.3 Tree Edit Distance (TED)

In computer science, trees are among the most important structures that are investigated and used. Examples of common areas that trees are being used in are tree automata, tree grammars, image analysis, computational biology [3, 67, 89, 98].

Tree edit distance was first proposed by Tai in 1979 [89]. Zhang and Shasha [98] improved the tree edit distance. Other researches have later proposed algorithms for tree edit distance [7, 12, 21, 48, 66, 67].

Pawlik and Augsten [66] introduced robust tree edit distance (RTED). Robust in RTED means it is independent of the tree shape and performs better in computation of sub-problems compared to previously proposed algorithms [67]. Pawlik and Augsten [67, 68] have made several improvements of RTED. Okundaye et al. [63] conducted a survey of different CBIR systems and observed that the RTED provides better retrieval results for syntactically generated images.

2.2.4 Summary of Mathematical Similarity Measures

In this section, we have outlined mathematical similarity measures. For this research, we use mathematical similarity measures to determine the similarity of the generated pictures. Spatial colour distribution descriptor is a compact composite descriptor which combines colour and spatial colour distribution information [6]. It was chosen to measure similarity mathematically for this study as it was identified to be efficient in finding similar pictures for syntactically generated pictures.

2.3 Perceptual Similarity

The problem with many image retrieval systems is analysing the relationship between perceptual similarity and approaches used in CBIR. Mathematically based similarity measures are capable of finding similar pictures, but people may not find those pictures similar. Humans judge the similarity of pictures by considering many features such as colour, semantics, luminance, texture and objects in the image [53, 60, 97, 99]. This shows that, even if humans do not think specific pictures are similar, it does not necessarily mean that they are not. This can also be the reason that different people have different opinions on the similarity of a given set of pictures. Most CBIR systems are based on one or more of these features. Even so, humans are the final judges of the similarity of a given picture to a set of similar pictures; it is, therefore, essential to determine if the generated structurally similar pictures correspond to a human perceptual similarity measure.

To understand visual perception, several researchers have tried to compare their findings of applying mathematical similarity measures to pictures with human perception in order to understand visual perception; in the case of colour, Kiranyaz et al. [47] have tried to model human colour perception. They observed that the human eye could not distinguish close colours well or identify a broad number of colours. Thus, they showed that humans only use a few outstanding colours to judge similarity. They presented a systematic approach to extract a perceptual (colour) descriptor and then proposed an efficient similarity metric to achieve the highest discrimination power possible for colour-based retrieval in general-purpose image databases.

Moreover, Yamamoto et al. [97] conducted an experiment to evaluate the correlation between their similarity function and human perception. They selected 20 query images with various representative arrangements of colours and regions to a 10,000 picture database and asked respondents to say if the retrieved image was relevant or not relevant. The researchers found that humans judge similarity based on different factors such as objects and colours in the picture. Hence humans may sometimes have different opinions

on the similarity of given pictures.

Furthermore, Okundaye et al. [65] conducted an online survey requiring respondents to arrange pictures generated by tree grammars in the order of similarity to a given image. Their focus was also on using the pictures in a visual password system which was to be used by humans. Their results showed that, although humans are generally in agreement about the similarity of images, they sometimes had different opinions.

The generated picture sets in our research are for use in visual password systems. Many people use these systems, and so measuring perceptual similarity, it is ideal for getting a sample size that represents a large population. For us to get enough sample size, we selected an online survey for this study. Moreover, Okundaye et al. [65] used an online survey after a comprehensive analysis of perceptual similarity for similar research. There are many benefits of an online survey compared to face to face interviews, focus groups or hardcopy questionnaires. Some of the benefits of an online survey are it is possible to get a wider audience, cheaper to administer and no need to follow the respondents physically.

Some negative aspects of a survey, in general, are dishonesty in answering questions due to privacy concerns and unanswered questions due to the length of questionnaires or boredom. To overcome dishonesty and unanswered questions, we have designed a short and anonymous survey (no personal details were taken about respondents). There are several limitations of an online survey. For example, respondents are unable to ask questions if they do not understand something. Moreover, in the case of multiple-choice questions, it may be difficult for them to express their opinions. Another challenge is that only people with access to internet and computer may participate. To overcome the first and second challenge, we have provided e-mail for participants to contacts us in case of any questions and opinions.

2.4 Visual Password Systems

Visual password systems use images as passwords for authentication. They are becoming popular now because humans are said to memorise images better than alphanumeric characters [2, 58]. The shortcomings of traditional alphanumeric passwords are that they are difficult to memorise, can easily be shared and the fact that many people write them down or use names of common things or people in their lives in order to remember the passwords [62, 64, 95]. All these factors make alphanumeric passwords less secure, hence visual passwords are thought to be a better alternative. According to Biddle et al. [2], visual password systems are categorised depending on how they are memorised as follows: recall, recognition and cued-recall. In the following section we discuss several types of visual password systems.

2.4.1 Recall-Based Visual Password Systems

In recall-based visual password systems, a person is required to remember information without cueing [2]. Recall requires a user to draw their password either on a blank canvas or on a grid. Examples of recall-based visual password systems are Draw-A-Secret (DAS) [41], Background Draw-A-Secret (BDAS) [22] and a doodle password (Passdoodle) [33]. DAS systems use simple pictures drawn in a grid. DAS does not depend on alphanumeric letters [41]. Figure 2.27 shows an example of a graphical password input on a 4×4 grid. The system records the password by the use of coordinate pairs by mapping the sequence and the order in which the stylus passes through them [41]. The drawback of recall-based visual password systems is severe shoulder-surfing as in most cases when the user enters a password, a shoulder-surfer can view the whole drawing on a single login [2]. An improvement on DAS, BDAS [22] was introduced to encourage users to create more complex passwords by adding a background image to DAS. Using a background image in BDAS was seen to improve the strength of passwords, reduce symmetry and to centre within password images [22].

Passdoodles are graphical passwords comprising of handwritten designs or text commonly drawn onto a touch screen device [33]. Passdoodles have a larger number of possible doodle passwords due to the use of additional characteristics such as pen colour, number of pen strokes and speed of drawing. Passdoodle uses a complex matching process without a visible grid [2]. Passdoodles are more difficult to crack than alphanumeric passwords considering the larger number of possible doodle passwords [33]. The weakness of Passdoodles is they are difficult to recall information due to lack of memory prompts or cues [2].

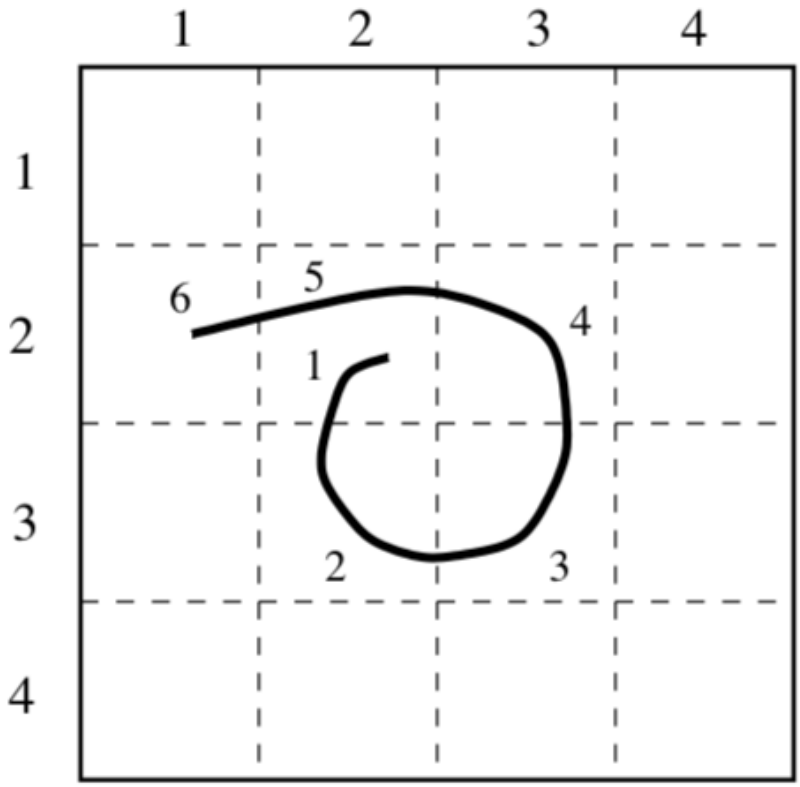


Figure 2.27: Example of input of DAS password on a 4 × 4 grid

2.4.2 Recognition-Based Visual Password Systems

Recognition requires a person to decide if the previously memorised information matches the presented information. Recognition has been seen as an easier memory task than recall [2]. In recognition-based visual password systems, user is required to remember selected images as passwords, and distinguish them from distractor images when authenticating. Recognition-based visual password systems is also affected by shoulder surfing due to the possibility of an attacker to spy user-selected pictures as they log in.

Examples of recognition-based visual password systems are Passfaces [74], Story [11] and Deja Vu [13]. Generally, in the Passfaces type of visual password system, faces are used as passwords [74]. A user password will be a selected number of faces. In order to authenticate into a system, the user has to select correct faces displayed in a $n \times n$ grid. For example, see Figure 2.28, a 3×3 grid containing both password and decoy images. If the user selects correct images in all instances, the system becomes accessible [58, 64, 74]. On the other hand, if a user selects a wrong image even once, the system allows finishing with authentication, then denies access.

The Passfaces type of visual password system has been shown to have a weakness of bias in choosing faces for passwords between male and female users [58, 62]. Other weaknesses of the Passfaces type of visual password system are that it requires searching in large databases for faces, dealing with copyright issues and a lack of similar pictures.

Picture grammars have recently been used to generate similar pictures for use in Passfaces type of visual password system to overcome bias and a lack of images to use as passwords [62, 64]. Okundaye et al. [64] developed a prototype Passfaces type of visual password system using picture grammars. This system dynamically generates password images removing the need for a large image database. The system also eliminates the need for high network bandwidth as there is no need to load images over the network. Picture grammars can generate many pictures for use at no cost. Moreover, picture grammars generate abstract pictures, hence minimising bias [62]. Figures 2.29–2.32 show some authentication steps in the prototype that uses picture grammars to generate its pictures taken from [62].

The system developed by Okundaye et al. [64] is made up of three subsystems, specifically the picture grammar generator, the mathematical similarity and the visual password systems. In the first authentication step in Figure 2.29, the user selects a picture associated with a grammar. The picture grammar generator reads grammar and produces sixteen images. Then the mathematical similarity finds the eight most similar images to the selected image. In the second authentication step, the nine pictures (chosen picture and the eight most similar) are then presented to the user to select their password image. Figure 2.30–2.32 shows how the pictures are

presented to the user. The selected picture becomes the password, while the remaining ones are the distractor images. The visual password system allows specifying how many password images should be chosen for each user with a minimum of three. The first and second authentication step is repeated until the specified number of password images in the visual password system have been selected. This information is then saved on the user profile. During login, the user will be presented with the nine pictures placed randomly in a 3×3 grid and asked to select their password image (example Figures 2.30–2.32). The process is repeated two more times. Access is granted only if the user selected the correct image at each authentication step. In this research, we want to improve the similarity of the pictures generated by picture grammars for use in visual password systems.

Story uses images as passwords like Passfaces [11]. In Story, a user must select images in a sequential manner, the order of images must be correct. For ease of memorising the passwords, users are advised to connect their images like a Story [2, 11]. In Deja Vu like Passfaces and Story, use images as passwords, but in it they used random abstract images as passwords [13].

2.4.3 Cued-Recall-Based Visual Password Systems

In cued-recall, a person is provided with an external cue to help remember information [2]. In cued-recall-based visual password systems a user is required to remember some parts of the image. Cued-recall-based visual password systems are easier to memorise than recall-based visual password systems [2]. An example of cued-recall-based visual password systems is Passpoint [2]. In Passpoint visual, users click on a sequence of points on one or more background images [2, 93]. The password is a user-selected sequence of points on a system provided image. To log in to the system, the user must click on selected points in the exact sequence. The image provides a cue to the location of the initially chosen click-points.

2.4.4 Summary of Visual Password Systems

In this section, we have discussed several visual password systems detailing how they work, their strength and weaknesses. In this research, we generate a finite number of similar pictures to be used in a Passfaces type of visual password systemİn the Passfaces type of visual password system, the user selects images as a password among a set of images. To authenticate into the system, the user in all instances has to select the correct images displayed in a $n \times n$ grid. Due to bias in selecting images in a Passfaces type of visual password systems, abstract images were seen as a better alternative. Moreover, Passfaces require searching in large databases for faces, dealing with copyright issues and lack of similar pictures. Several researchers have

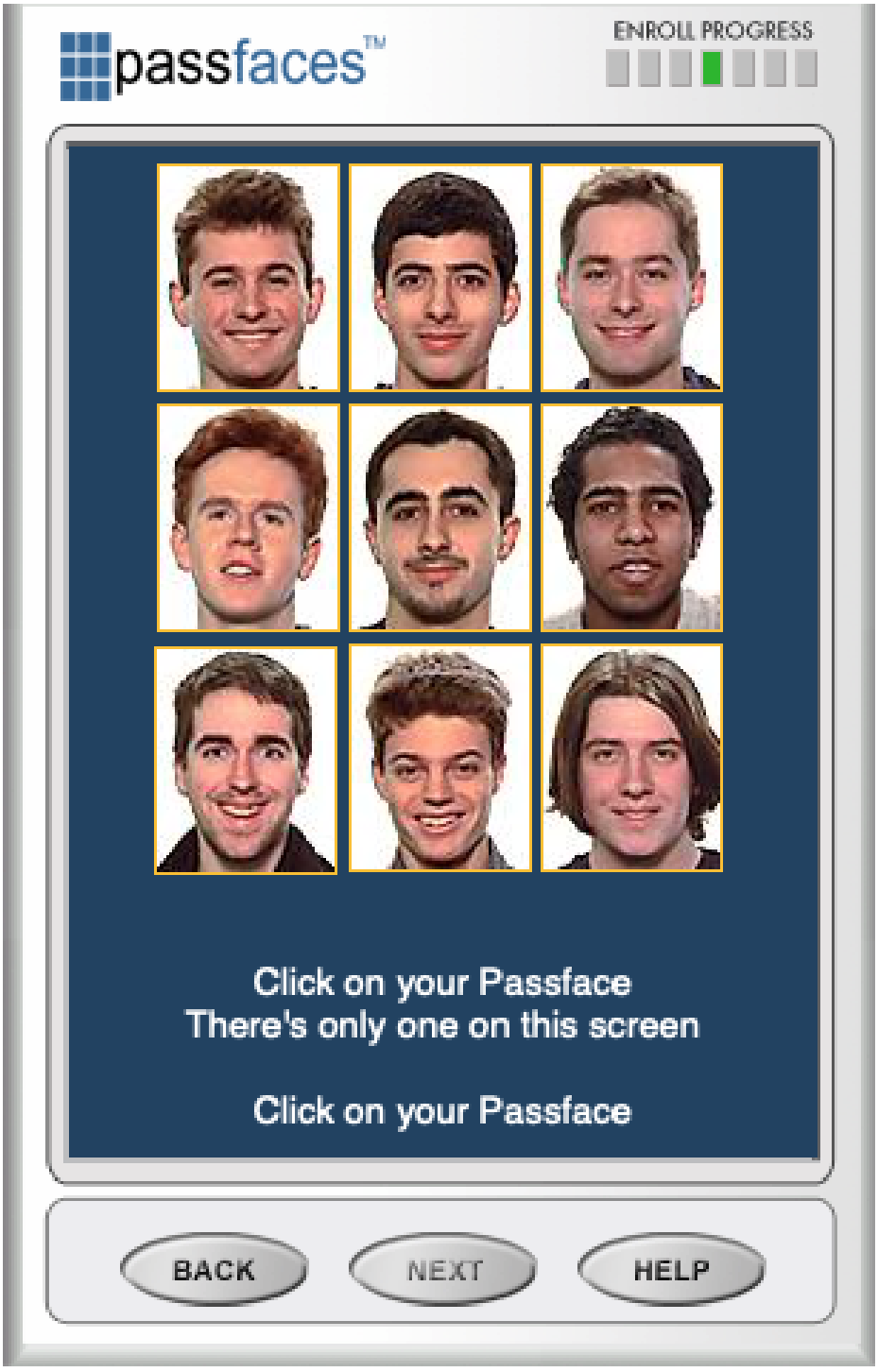


Figure 2.28: Example of Passfaces system on a 3 × 3 grid containing users password and distractor images

Figure 2.29: Example of first authentication step

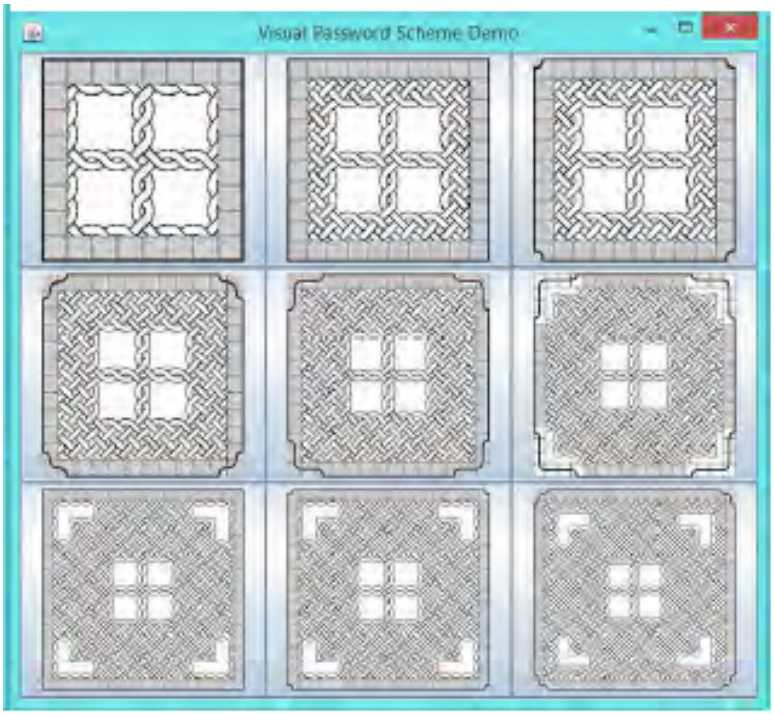


Figure 2.30: Example of second authentication step

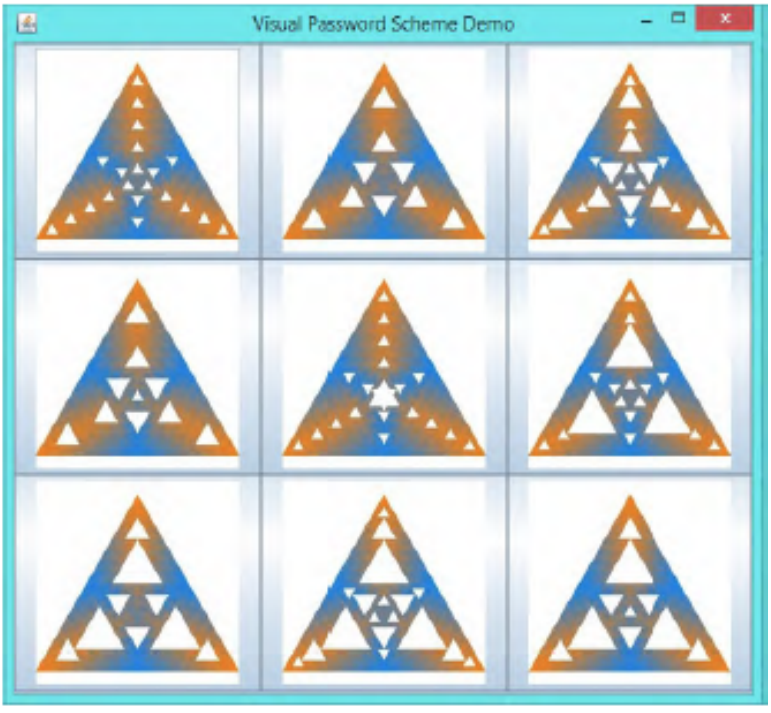


Figure 2.31: Example of third authentication step

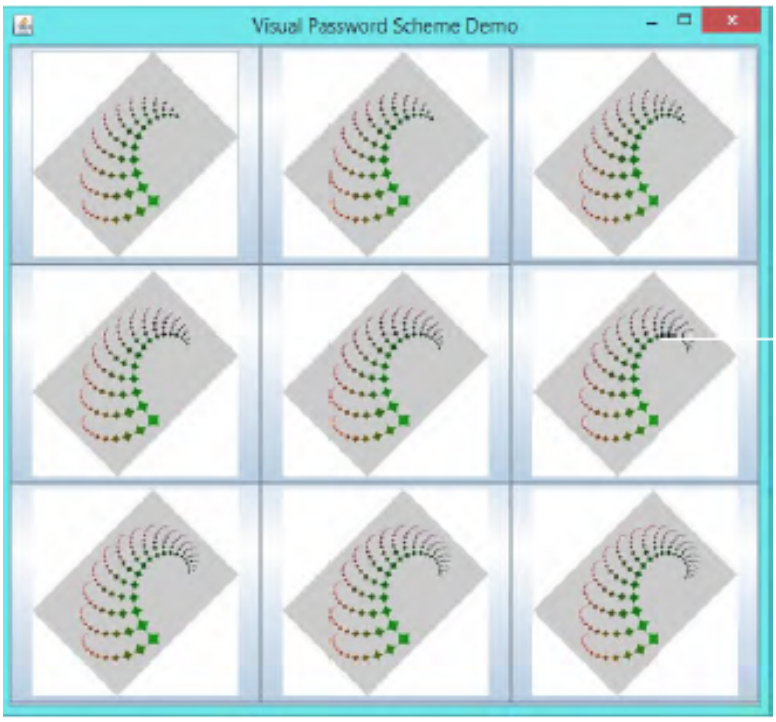


Figure 2.32: Example of fourth authentication step

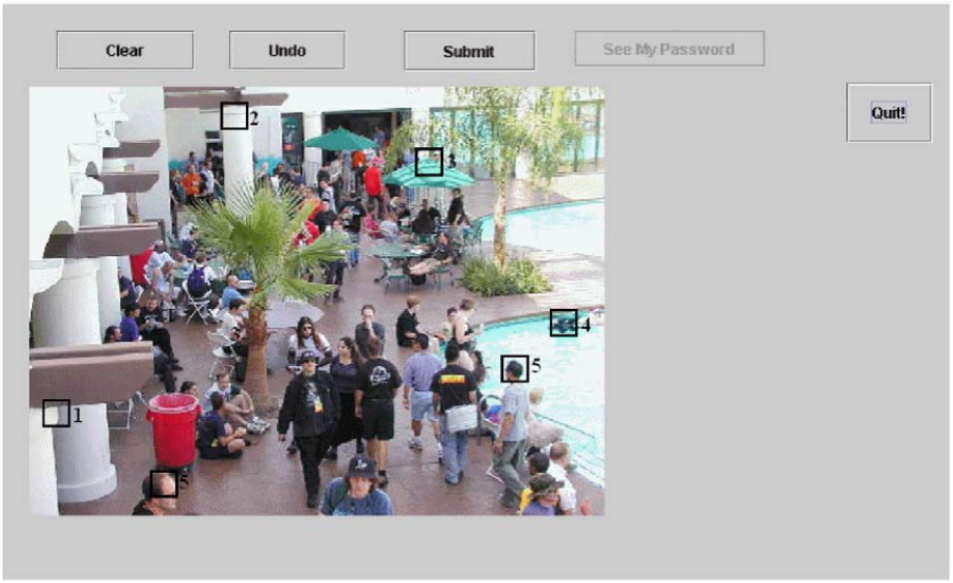


Figure 2.33: Example of user Passpoint password with click order displayed

proposed other image types to be used for recognition-based visual password system like abstract images, random arts and pictures of nature scenes [13, 64].

Picture grammars were also used to generate similar pictures for the Passfaces type of visual password system in recent research by Okundaye et al. [64]. Picture grammars can generate many pictures for use at no cost. Furthermore, using picture grammars eliminates the need for searching for similar pictures in databases and dealing with copyright issues. Moreover, picture grammars generate abstract pictures, hence minimising bias. In this research, we want to improve the similarity of the pictures generated by picture grammars for use in the Passfaces type of visual password system.

2.5 Conclusion

This chapter discussed different syntactic picture generating mechanisms that demonstrate the problems with the picture grammars currently available and motivated why RCPGs were considered for generation of structurally similar pictures. RCPGs are context-free grammars with regulated rewriting. Regulated rewriting in RCPGs makes them easier to control the application of rules and hence it will be easier for us to control the produced picture sets. Moreover, the literature review has motivated us to define a BCPG (which is discussed in more detail in Chapter 3) that have great potential in the generation of pictures, particularly structurally similar pictures.

Furthermore, we discussed several mathematical similarity measures for measuring the mathematical similarity of pictures based on texture, colour and tree edit distance and provided the motivation for choosing the SpCD for our research. SpCD was chosen to measure similarity mathematically for this study as it was identified to be efficient in finding similar pictures for syntactically generated pictures [63].

We then looked at different approaches by several researchers regarding perceptual similarity. Conducting an online survey to determine perceptual similarity was seen appropriate for this research due to its potential of obtaining many reviews. Lastly, we discussed previous research on visual password systems. The research on visual password systems assisted us in developing the methods of similar pictures, in terms of structure and number of the generated pictures.

Chapter 3

Bag Context Picture Grammars

3.1 Introduction

This chapter defines bag context picture grammars and shows how we can generate pictures using them. The definition of BCPGs is based on the definition of random context picture grammars [92] and bag context tree grammars [18] discussed in the previous chapter. This chapter is an extended version of the paper published in the Journal of Computer Languages [25].

In the previous chapter, we have studied different picture grammar types with the aim of learning their strengths, weaknesses and finding suitable picture grammar types for the generation of structurally similar pictures. Picture grammar types with the ability to produce variations in pictures are more appropriate for the generation of similar pictures, but also it should be easy to control the application of rules with them. For this reason, RCPGs were chosen for this research [92].

Although RCPGs use regulated rewriting in the application of rules, when it comes to the practical application of RCPGs, there are drawbacks. For example, when generating a set of pictures, one often requires the level of refinement to be between predetermined bounds: if the level is too low, the picture is of little interest, if the level is too high, the human eye cannot detect details anymore. This can be enforced with random context, but generally requires a large number of variables and production rules. Moreover, when the bounds are changed, the grammar has to be rewritten. For this reason, so-called bag context was introduced for tree grammars [18].

In this chapter, we introduce bag context picture grammars. BCPGs can be viewed as a special case of bag context tree grammars, in particular, as tree grammars where each tree has the characteristic that each node has n^2 children, where $n = 1, 2, \ldots$. BCPGs are also context-free grammars, but the application of a rule is regulated by a k-tuple of integers, the bag, which

changes during the derivation of a picture. A rule in the grammar can only be applied if the bag, at that point, is within the scope defined by the lower and upper limits of the rule. When a rule is applied, it causes the bag values to change by adding the bag adjustment, which is part of the rule.

We investigate some aspects of BCPGs. In particular, we compare RCPGs and BCPGs. We show that any RCPG can be written as a BCPG. We consider this process for three sub-classes of RCPGs and in each case, illustrate it with an example.

For tree grammars, it was shown that bag context is strictly more powerful than random context [18, Theorem 5.1]. In this research, we do not continue that theoretical investigation. Rather, we consider one set of pictures and discuss why it is significantly easier to write and modify a BCPG that generates the set than an RCPG. One reason for this is that, in random context, it is not possible to determine how many of the permitting or forbidding variables are present in the developing picture, since only the presence or absence of a variable matters. In bag context, however, it is possible to keep track of the number of variables in the bag. Another reason is that, in bag context, the application of a rule does not necessarily depend on the existence or absence of variables. Other conditions for a rule to be applicable may be defined, for example, the number of times a particular rule has been applied. This information can also be kept in the bag.

The remainder of this chapter is organised as follows. Section 3.2 presents the definitions of BCPGs and all relevant concepts. In Section 3.3 we show that any RCPG can be written as a BCPG. We consider this process for four sub-classes of RCPGs and in each case, illustrate it with an example. In Section 3.4, we consider two sets of pictures and discuss why it is easier to write BCPGs that generate a variation on the sets, and modify those BCPGs again, should we wish to, than to do the same to the RCPGs that generate the original picture sets. Section 3.5 presents the conclusion and future work.

3.2 Definitions

In this section, we introduce notation and definitions. In particular, we define bag context picture grammars. Many definitions are from [18, 24, 92] and have been adapted as necessary.

3.2.1 Bag Context Picture Grammars

Bag context picture grammars generate pictures using productions of the form in Figure 3.1, where A is a variable, x_{11}, $x_{12}, \ldots, x_{mm}$ are variables or terminals for $m \in \mathbb{N}_+$, and λ, μ and δ are k-tuples for some $k \in \mathbb{N}_+$. The

interpretation is as follows: if a developing picture contains a square labelled A and if the bag is within the range defined by the lower limit λ and upper limit μ of the rule, then the square labelled A may be divided into equal squares with labels $x_{11}, x_{12}, \ldots, x_{mm}$ and the bag adjusted with δ.

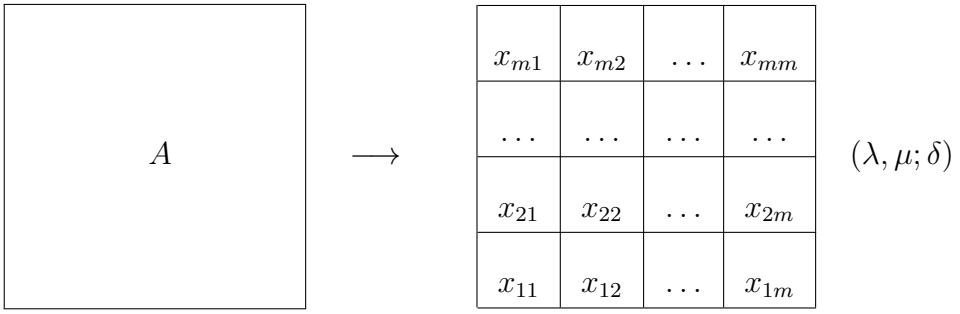


Figure 3.1: Production in BCPG

We denote a square by a lowercase Greek letter, for example, (A, α) denotes a square α labelled A. If α is a square, then $\alpha_{11}, \alpha_{12}, \ldots, \alpha_{mm}$ denote the equal-sized sub-squares into which α can be divided, for example, if a square is divided into four equal-sized sub-squares, then the sub-squares α_{11}, α_{12}, α_{21} and α_{22} denote the bottom left, bottom right, top left and the top right sub-squares respectively.

Definition 3.1. A bag context picture grammar $G = (V_{\mathrm{N}}, V_{\mathrm{T}}, P, (S, \sigma), I, \beta_0)$ has a finite alphabet V of labels, consisting of disjoint subsets V_{N} of variables and V_{T} of terminals. P is a finite set of production rules. There is an initial labelled square (S, σ) with $S \in V_{\mathrm{N}}$. Finally, I denotes a finite bag index set and $\beta_0 \in \mathbb{Z}^I$ the initial bag.

A rule in P is of the form $A \to [x_{11}, x_{12}, \ldots, x_{mm}]\ (\lambda, \mu; \delta)$, $m \in \mathbb{N}_+$, where $A \in V_{\mathrm{N}}$, $\{x_{11}, x_{12}, \ldots, x_{mm}\} \subseteq V$, $\lambda, \mu \in \mathbb{Z}^I_\infty$, and $\delta \in \mathbb{Z}^I$. The k-tuples λ and μ are the lower and upper limits respectively, while δ is the bag adjustment.

For simplicity, we write a rule of the form $A \to [x_{11}, x_{12}, \ldots, x_{mm}]$ $(-\infty, \infty; 0)$ as $A \to [x_{11}, x_{12}, \ldots, x_{mm}]$.

Definition 3.2. A pictorial form is any finite set of non-overlapping labelled squares in the plane. The size of a pictorial form Π is the number of squares contained in it, denoted $|\Pi|$. If Π is a pictorial form, we denote by $l(\Pi)$ the set of labels used in Π.

Definition 3.3. Let $G = (V_{\mathrm{N}}, V_{\mathrm{T}}, P, (S, \sigma), I, \beta_0)$ be a BCPG, Π and Γ pictorial forms, and $\beta, \beta' \in \mathbb{Z}^I_\infty$. Then we write $(\Pi, \beta) \Longrightarrow (\Gamma, \beta') \in \Pi \times \mathbb{Z}^I_\infty$. There is a derivation step from (Π, β) to (Γ, β') if there is a production

$A \to [x_{11}, x_{12}, \ldots, x_{mm}]$ $(\lambda, \mu; \delta)$ in P, Π contains a labelled square (A, α), $\lambda \leq \beta \leq \mu$, $\Gamma = (\Pi \setminus \{(A, \alpha)\}) \cup \{(x_{11}, \alpha_{11}), (x_{12}, \alpha_{12}), \ldots, (x_{mm}, \alpha_{mm})\}$, and $\beta' = \beta + \delta$. We denote the derivation step by $(\Pi, \beta) \Longrightarrow (\Gamma, \beta')$. As usual, $\Longrightarrow^*$ denotes the reflexive transitive closure of $\Longrightarrow$.

Definition 3.4. The (bag context) picture gallery generated by a BCPG $G = (V_{\mathrm{N}}, V_{\mathrm{T}}, P, (S, \sigma), I, \beta_0)$ is the set $\mathcal{G}(G) = \{\Phi \mid (\{(S, \sigma)\}, \beta_0) \Longrightarrow^* (\Phi, \beta)\,,$ for $\beta \in \mathbb{Z}_\infty^I$ and $l(\Phi) \subseteq V_{\mathrm{T}}\}$. An element of $\mathcal{G}(G)$ is called a picture.

Definition 3.5. Let Φ be a picture in the square σ. For any $m \in \mathbb{N}_+$, let σ be divided into equal-sized sub-squares, say $\sigma_{11}, \sigma_{12}, \ldots, \sigma_{mm}$. A sub-picture Ψ of Φ is any subset of Φ that fills a square σ_{ij}, with $i, j \in [m]$, that is, the union of all the squares in Ψ is the square σ_{ij}.

In the following, we give an example of a BCPG and we demonstrate how they work.

Example 3.1. Consider the gallery $\mathcal{G}_{\mathrm{carpet}}$. Four pictures in $\mathcal{G}_{\mathrm{carpet}}$ are shown in Figure 3.2, where Figure 3.2a is the smallest picture in this gallery, Figure 3.2b the second smallest, and so forth.

$\mathcal{G}_{\mathrm{carpet}}$ consists of a variation on the sequence of pictures approximating the Sierpiński carpet [1]. The smallest picture (Figure 3.2a) is divided into nine equal sub-squares. The sub-squares in the corners and in the centre are filled with purple circles on a grey background, while the remaining sub-squares are filled with white circles on a grey background. In the second smallest picture (Figure 3.2b), the white circles are as in Figure 3.2a. Each remaining square is filled with a modified copy of Figure 3.2a, where the modification consists of having green circles instead of white ones. In the third smallest picture (Figure 3.2c), the white and green circles are as in Figure 3.2b, while each remaining square is filled with a copy of Figure 3.2a. This pattern continues from one picture in the sequence to the next.

$\mathcal{G}_{\mathrm{carpet}}$ is created by the BCPG $G_{\mathrm{carpet}} = (V_{\mathrm{N}}, V_{\mathrm{T}}, P, (S, \sigma), [5]\,, (1, 0, 0, 0, 0))$, where $V_{\mathrm{N}} = \{S, T, U, F\}$, $V_{\mathrm{T}} = \{w, b, g\}$, and P is the set of rules in Figure 3.3. Terminals w, b and g represent white, purple and green circles, respectively.

The bag in G_{carpet} has five positions. The first four positions count the number of occurrences of S, T, U and F respectively, in the developing picture. The fifth bag position determines whether the circles in a level of refinement are white or green. It alternates between 0 and 1. It has the value 0 in the first refinement; therefore there are white–not green–circles in Figure 3.2a.

The bag context regulates the application of the rules. For example, in Rule 3.1, the third bag position of the upper limit is 0. Therefore, this rule can only be applied if there is no U present in the pictorial form.

The derivation of any given picture starts with the initial square labelled S. At this point, Rule 3.1 is applicable, since $(1,0,0,0,0) \leq (1,0,0,0,0) \leq (\infty,\infty,0,\infty,0)$, but Rule 3.2 is not, since $(1,0,0,0,1) \not\leq (1,0,0,0,0) \leq (\infty,\infty,0,\infty,\infty)$. Rule 3.1 divides the initial square into nine equal-sized sub-squares, each labelled with the variable T or the terminal w.

This pictorial form can derive a picture or a more refined pictorial form. A picture is derived when any T produces an F (Rule 3.5). The bag context ensures that every remaining T then also becomes an F by the same rule. Once this has been done, each F produces a b (Rule 3.8).

A more refined pictorial form is derived when any T it produces a U (Rule 3.3). Every remaining T then also becomes a U by either Rule 3.3 or Rule 3.4. This is done by alternating between the two rules. When Rule 3.3 is applied, the last bag position gets the value 1, and the application of Rule 3.4 resets the value of the last bag position to 0. Rule 3.3 is applied in the last T in the pictorial form; therefore green–not white–circles will be used in the next round.

Once this has been done, each U is replaced by an S (Rule 3.7).

At this point, Rule 3.2, but not Rule 3.1, is applicable. Rule 3.2 divides one square labelled S into nine equal-sized sub-squares, each labelled with the variable T or the terminal g. The bag context ensures that every remaining square labelled S is sub-divided in the same manner by the same rule.

As above, this pictorial form can derive a picture or a more refined pictorial form, and the decision is made by the derivation of any T. To produce a picture, the procedure is as above. If a more refined pictorial form is derived, any T produces a U by Rule 3.4. The bag context ensures that Rule 3.4, but not Rule 3.3, is applicable. Every remaining T then also becomes a U by either Rule 3.3 or Rule 3.4. This is done by alternating between the two rules. Once this has been done, each U is replaced by an S (Rule 3.7).

This process can be repeated an arbitrary number of times.

Table 3.1 illustrates a derivation of Figure 3.2a. In the first column, we give the sequence of the pictorial forms in the derivation. In the second column we illustrate the evolution of the bag during this derivation. We provide the rule that is applied to the pictorial form to generate the pictorial form in the next row in the third column.

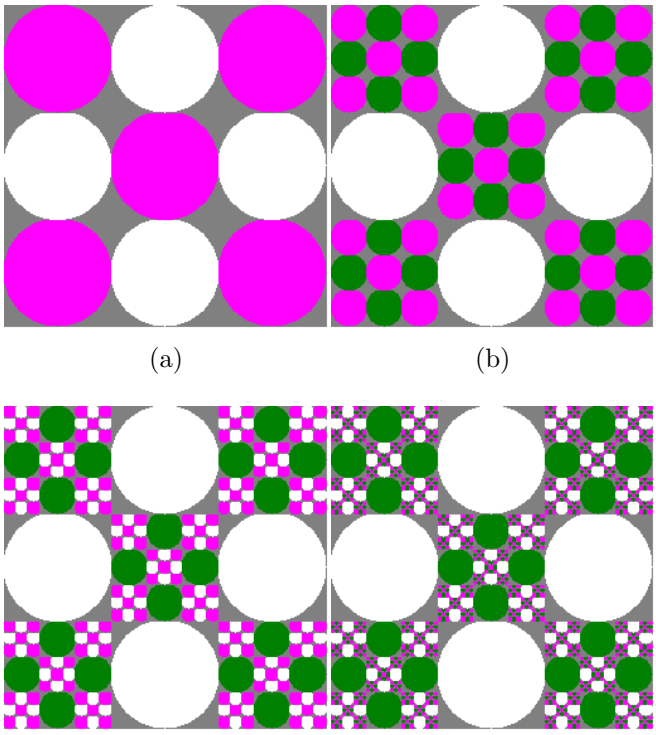


Figure 3.2: A selection of pictures in $\mathcal{G}_{\text{carpet}}$

$$
\begin{aligned}
S \to\ & [T,w,T,w,T,w,T,w,T]\,((1,0,0,0,0)\,,(\infty,\infty,0,\infty,0)\,; \\
& \quad (-1,5,0,0,0))\ \mid && (3.1)\\
& [T,g,T,g,T,g,T,g,T]\,((1,0,0,0,1)\,,(\infty,\infty,0,\infty,\infty)\,; \\
& \quad (-1,5,0,0,0)) && (3.2)\\
T \to\ & U\,((0,1,0,0,0)\,,(0,\infty,\infty,0,0)\,;(0,-1,1,0,1))\ \mid && (3.3)\\
& U\,((0,1,0,0,1)\,,(0,\infty,\infty,\infty)\,;(0,-1,1,0,-1))\ \mid && (3.4)\\
& F\,((0,1,0,0,0)\,,(0,\infty,0,0,\infty)\,;(0,-1,0,1,0))\ \mid && (3.5)\\
& b\,((0,1,0,1,0)\,,\infty;(0,-1,0,0,0)) && (3.6)\\
U \to\ & S\,((0,0,1,0,0)\,,(\infty,0,\infty,\infty,\infty)\,;(1,0,-1,0,0)) && (3.7)\\
F \to\ & b\,((0,0,0,1,0)\,,(\infty,0,\infty,\infty,\infty)\,;(0,0,0,-1,0)) && (3.8)
\end{aligned}
$$

Figure 3.3: Rules for grammar G_{carpet}

Table 3.1: Derivation of Figure 3.2a

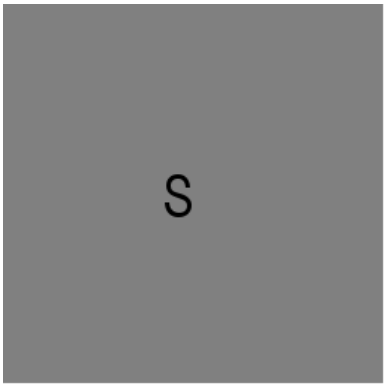
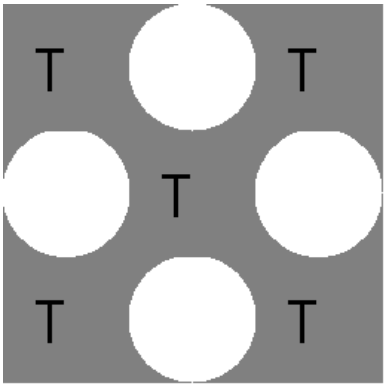
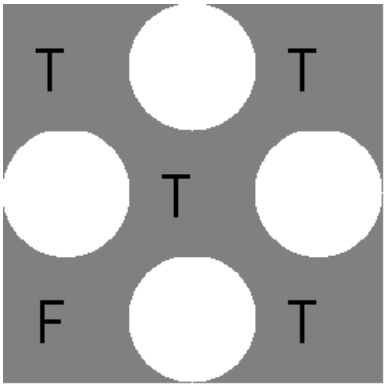
Pictorial form	**Current bag**	**Next rule**
	$(1, 0, 0, 0, 0)$	$S \rightarrow$ $[T, w, T, w, T, w, T, w, T]$ $((1, \ 0, \ 0, \ 0, \ 0),$ $(\infty, \ \infty, \ 0, \ \infty, \ 0);$ $(-1, \ 5, \ 0, \ 0, \ 0))$
	$(0, 5, 0, 0, 0)$	$T \rightarrow F$ $((0, \ 1, \ 0, \ 0, \ 0),$ $(0, \ \infty, \ 0, \ 0, \ \infty);$ $(0, \ -1, \ 0, \ 1, \ 0))$
	$(0, 4, 0, 1, 0)$	$T \rightarrow b$ $((0, \ 1, \ 0, \ 1, \ 0),$ $(\infty, \ \infty, \ \infty, \ \infty, \ \infty);$ $(0, \ -1, \ 0, \ 0, \ 0))$

Table 3.1 continued from previous page

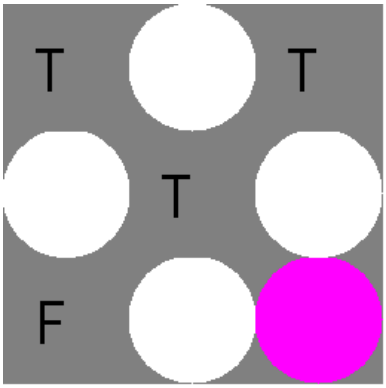
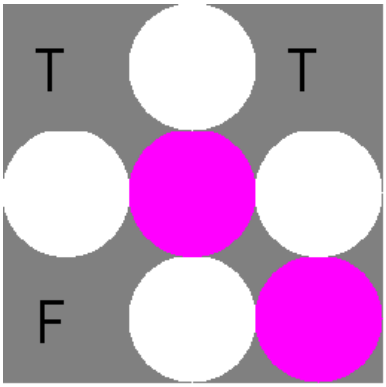
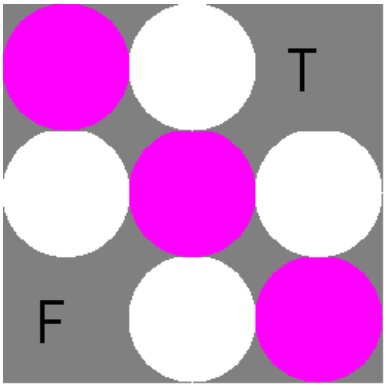
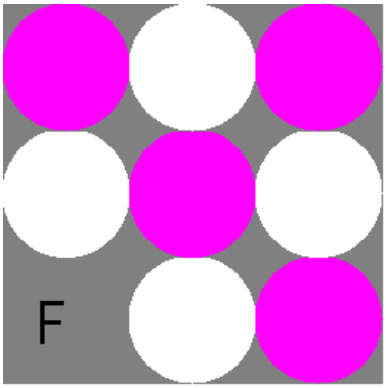
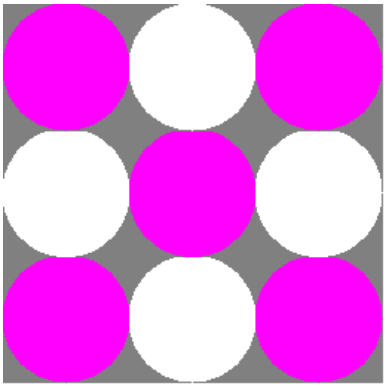
Pictorial form	Current bag	Next rule
	$(0, 3, 0, 1, 0)$	$T \to b$ $((0, \ 1, \ 0, \ 1, \ 0),$ $(\infty, \ \infty, \ \infty, \ \infty, \ \infty);$ $(0, \ -1, \ 0, \ 0, \ 0))$
	$(0, 2, 0, 1, 0)$	$T \to b$ $((0, \ 1, \ 0, \ 1, \ 0),$ $(\infty, \ \infty, \ \infty, \ \infty, \ \infty);$ $(0, \ -1, \ 0, \ 0, \ 0))$
	$(0, 1, 0, 1, 0)$	$T \to b$ $((0, \ 1, \ 0, \ 1, \ 0),$ $(\infty, \ \infty, \ \infty, \ \infty, \ \infty);$ $(0, \ -1, \ 0, \ 0, \ 0))$

Table 3.1 continued from previous page

Pictorial form	Current bag	Next rule
	$(0, 0, 0, 1, 0)$	$F \to b$ $((0, 0, 0, 1, 0),$ $(\infty, 0, \infty, \infty, \infty);$ $(0, 0, 0, -1, 0))$
	$(0, 0, 0, 0, 0)$	none

3.3 Rewriting an RCPG as a BCPG

In this section, we show that any random context picture grammar can be written as a bag context picture grammar.

3.3.1 Notation

We first define some notation used in this section. Let $G = (V_{\mathrm{N}}, V_{\mathrm{T}}, P, (S, \sigma))$ be an RCPG. Assume $V_{\mathrm{N}} = \{A_1, A_2, \ldots, A_m\}$, and that the order of elements in V_{N} is arbitrary but fixed. For rule $A \to [x_1, \ldots, x_m]$ $(\mathcal{P}; \mathcal{F})$, we denote by cnt the multiplicity of elements of V_{N} in the lefthand side, the righthand side and the random context of the rule. In particular,

- $\mathrm{cnt}(A)$ is the m-tuple in which the component corresponding to A is set to 1, whereas all other components are set to 0,
- $\mathrm{cnt}([x_1, \ldots, x_m])$ is the m-tuple $(n_1, n_2, \ldots, n_m)$ such that n_i is equal to the number of occurrences of A_i in $[x_1, \ldots, x_m]$, and

- $\operatorname{cnt}(\mathcal{P}) = \sum_{A \in \mathcal{P}} \operatorname{cnt}(A)$, respectively $\operatorname{cnt}(\mathcal{F}) = \sum_{A \in \mathcal{F}} \operatorname{cnt}(A)$.

The following lemma is based on [18, Lemma 5.1].

Lemma 3.1. Let $G = (V_{\mathrm{N}}, V_{\mathrm{T}}, P, (S, \sigma))$ be a RCPG. Then there is an equivalent BCPG $G' = (V_{\mathrm{N}}, V_{\mathrm{T}}, P', (S, \sigma), I, \beta_0)$ such that $\mathcal{G}(G) = \mathcal{G}(G')$.

Proof. Without loss of generality, assume that $V_{\mathrm{N}} = \{A_1, A_2, \ldots, A_m\}$. The bag index set $I = [m]$ records, at all times during a derivation, the number of occurrences of A_i in the ith position of the bag.

Let $\beta_0 = \operatorname{cnt}(S)$. For every random context rule $A \to [x_1, \ldots, x_m]$ $(\mathcal{P}; \mathcal{F})$ define the bag context rule $A \to [x_1, \ldots, x_m]$ $(\lambda, \mu; \delta)$, where

- $\lambda = \operatorname{cnt}(A) + \operatorname{cnt}(\mathcal{P})$,
- $\mu = \operatorname{cnt}(A) + \infty \cdot \operatorname{cnt}(V_{\mathrm{N}} \setminus \mathcal{F})$ (where, by convention, $\infty \cdot 0 = 0$), and
- $\delta = \operatorname{cnt}([x_1, \ldots, x_m]) - \operatorname{cnt}(A)$.

Clearly, $\mathcal{G}(G) = \mathcal{G}(G')$. □

The following example gives an RCPG and the corresponding BCPG.

Example 3.2. Consider the gallery $\mathcal{G}_{\text{random}}$ which consists of the sequence of pictures of the Sierpiński carpet. Three pictures in $\mathcal{G}_{\text{random}}$ are shown in Figure 3.4, where Figure 3.4a is the smallest picture in the gallery, Figure 3.4b is the second smallest picture in the gallery, and so forth.

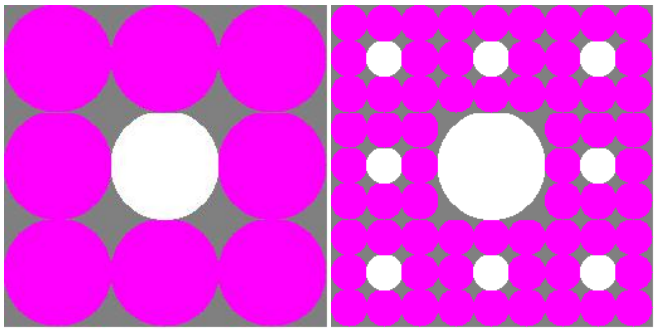
The smallest picture (Figure 3.4a) is divided into nine equal-sized sub-squares. The sub-square in the middle is filled with a white circle on a black background, while the remaining sub-squares are filled with grey circles on a black background. In the second smallest picture (Figure 3.4b), the white circle is as in Figure 3.4a. Each remaining square is filled with a copy of Figure 3.4a. In the third smallest picture (Figure 3.4c), the white circles are as in Figure 3.4b, while each remaining square is filled with a copy of Figure 3.4a. This pattern continues from one picture in the sequence to the next.

$\mathcal{G}_{\text{random}}$ is generated by the RCPG $G_{\text{random}} = (V_{\mathrm{N}}, V_{\mathrm{T}}, P, (S, \sigma))$ where $V_{\mathrm{N}} = \{S, T, U, F\}$, $V_{\mathrm{T}} = \{w, b\}$ and P is the set of rules in Figure 3.5.

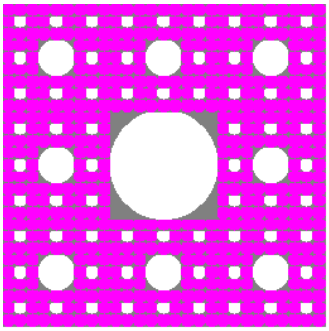
The corresponding BCPG is $G_{\text{random:bag}} = (V_{\mathrm{N}}, V_{\mathrm{T}}, P', (S, \sigma), [4], (1, 0, 0, 0))$ where $V_{\mathrm{N}} = \{S, T, U, F\}$, $V_{\mathrm{T}} = \{w, b\}$ and P' is the set of rules in Figure 3.6. The terminals w and b represent a white circle and a grey circle, respectively.

The bag positions 1 to 4 count the occurrences of the variables S, T, U and F respectively. Consider Rule 3.15. It can only be applied if the third position in both the lower and upper limits equals 0, that is, if there is no occurrence of the variable U in the pictorial form. Similarly, Rule 3.16 can

only be applied if there is no S or F in the pictorial form. On the other hand, Rule 3.17 can only be applied if the fourth position in the lower limit is at least 1, that is, if there is at least one occurrence of T in the pictorial form.



(a) (b)



(c)

Figure 3.4: A selection of pictures in $\mathcal{G}_{\text{random}}$

$$S \rightarrow [T,T,T,T,w,T,T,T,T]\,(\emptyset;\{U\}) \tag{3.9}$$
$$T \rightarrow U\,(\emptyset;\{S,F\}) \mid \tag{3.10}$$
$$F\,(\emptyset;\{S,U,F\}) \mid \tag{3.11}$$
$$b\,(\{F\};\emptyset) \tag{3.12}$$
$$U \rightarrow S\,(\emptyset;\{T\}) \tag{3.13}$$
$$F \rightarrow b\,(\emptyset;\{T\}) \tag{3.14}$$

Figure 3.5: Rules for grammar G_{random}

Next we demonstrate how to rewrite RPCPGs, RFCPGs and CFPGs as BCPGs. This is important for our research as in some instances, we

$$
\begin{aligned}
S \rightarrow\ & [T,T,T,T,w,T,T,T,T]\,((1,0,0,0)\,,(\infty,\infty,0,\infty)\,; \\
& \qquad (-1,8,0,0)) && (3.15)\\
T \rightarrow\ & U\,((0,1,0,0)\,,(0,\infty,\infty,0)\,;(0,-1,1,0))\ \mid && (3.16)\\
& F\,((0,1,0,0)\,,(0,\infty,0,0)\,;(0,-1,0,1))\ \mid \\
& b\,((0,1,0,1)\,,\infty;(0,-1,0,0)) && (3.17)\\
U \rightarrow\ & S\,((0,0,1,0)\,,(\infty,0,\infty,\infty)\,;(1,0,-1,0)) \\
F \rightarrow\ & b\,((0,0,0,1)\,,(\infty,0,\infty,\infty)\,;(0,0,0,-1))
\end{aligned}
$$

Figure 3.6: Rules for grammar $G_{\text{random:bag}}$

may want to use grammars that have been written in different subclasses of RCPGs.

3.3.2 The Special Case of RPCPGs

In this section, we consider random permitting context picture grammars. We derive the special case of Lemma 3.1 for this class of grammars where the forbidding context in every rule is empty. We give an example of grammar that demonstrates how RPCPG can be rewritten as BCPG.

Lemma 3.2. Let $G = (V_{\mathrm{N}}, V_{\mathrm{T}}, P, (S,\sigma))$ be an RPCPG. Let the order of the elements in V_{N} be arbitrary but fixed. Then there is a BCPG $G' = (V_{\mathrm{N}}, V_{\mathrm{T}}, P', (S,\sigma), I, \beta_0)$ such that $\mathcal{G}(G') = \mathcal{G}(G)$. In G', $\beta_0 = \mathrm{cnt}(S)$. Moreover, for every rule $A \rightarrow [x_1,\ldots,x_m]\ (\mathcal{P};\mathcal{F}) \in P$, there is a rule $A \rightarrow [x_1,\ldots,x_m]\ (\lambda,\mu;\delta) \in P'$, with

- $\lambda = \mathrm{cnt}(A) + \mathrm{cnt}(\mathcal{P})$,
- $\mu = \mathrm{cnt}(A) + \infty \cdot \mathrm{cnt}(V_{\mathrm{N}} \backslash \mathcal{F}) = \mathrm{cnt}(A) + \infty \cdot \mathrm{cnt}(V_{\mathrm{N}}) = \mathrm{cnt}(A) + \infty = \infty$, and
- $\delta = \mathrm{cnt}([x_1,\ldots,x_m]) - \mathrm{cnt}(A)$. □

The following example was adapted from Ewert [23], Ewert and van der Walt [27], provides an RPCPG and the corresponding BCPG.

Example 3.3. Consider the gallery $\mathcal{G}_{\text{permit}}$. Four pictures in $\mathcal{G}_{\text{permit}}$ are shown in Figure 3.7, where Figure 3.7a is the smallest picture in this gallery, Figure 3.7b the second smallest, and so forth. The smallest picture (Figure 3.7a) is divided into four equal-sized sub-squares. Each sub-square consists of the following image: in the centre there are four white circles on

a grey background, while the surrounding area is filled with black circles on a grey background. For the sake of simplicity, let us call this image π. Then, in general, every picture in $\mathcal{G}_{\text{permit}}$ has the following structure: the top left and bottom right quarters of a picture each consist of π. The top right quarter may be divided into four and π placed in each quarter; this may be recursively repeated for the top right quarter of the resulting sub-picture. The bottom left quarter of the picture must mirror the sub-division of the top right.

$\mathcal{G}_{\text{permit}}$ is generated by RCPG $G_{\text{permit}} = (V_{\text{N}}, V_{\text{T}}, P, (S, \sigma))$, where $V_{\text{N}} = \{S, A, B, A', B', F, C\}$, $V_{\text{T}} = \{w, b\}$, and P is the set of rules in Figure 3.8.

The terminal w represents a white circle on a grey background, while the terminal b represents a black circle on the same background. G_{permit} is a simple variation on the grammar in [28, Example 2.2].

Let the set of the elements in $V_{\text{N}} = \{S, A, B, A', B', F, C\}$, then the corresponding BCPG is $G_{\text{permit:bag}} = (V_{\text{N}}, V_{\text{T}}, P', (S, \sigma), [7], (1, 0, 0, 0, 0, 0, 0))$, where $V_{\text{T}} = \{w, b\}$, and P' is the set of rules in Figure 3.9

The bag positions 1 to 7 count the occurrences of S, A, B, A', B', F and C respectively. Consider Rules 3.18 and 3.19. Each can only be applied if the second and third position in the lower limit is at least 1, that is, if there is at least one occurrence of each of the variables A and B in the pictorial form.

3.3.3 The Special Case of RFCPGs

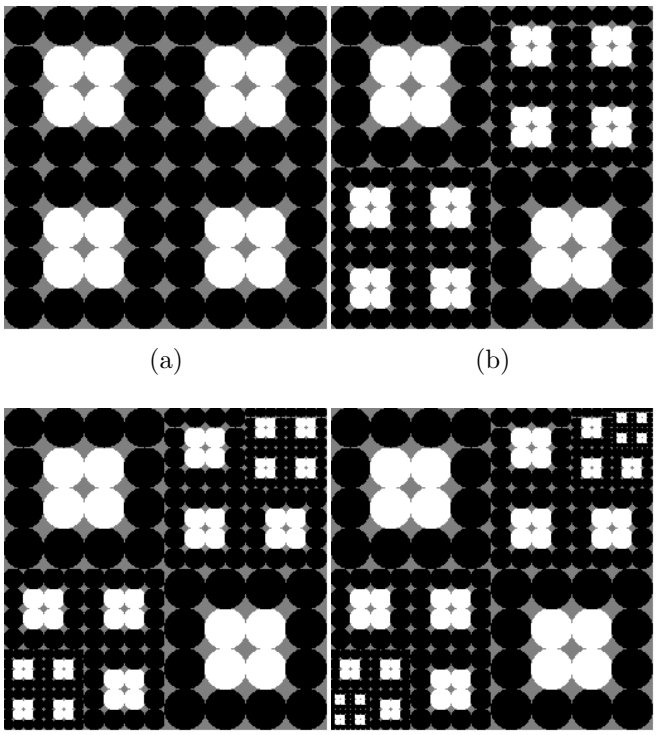
In this section, we consider random forbidding context picture grammars. We derive the special case of Lemma 3.1 for this class of grammars where the permitting context in every rule is empty. We give an example of grammar that demonstrates how RFCPG can be rewritten as BCPG.

Lemma 3.3. Let $G = (V_{\text{N}}, V_{\text{T}}, P, (S, \sigma))$ be an RFCPG. Let the set of the elements in V_{N} be arbitrary but fixed. Then there is a BCPG $G' = (V_{\text{N}}, V_{\text{T}}, P', (S, \sigma), I, \beta_0)$ such that $\mathcal{G}(G') = \mathcal{G}(G)$. In G', $\beta_0 = \text{cnt}(S)$. Moreover, for every rule $A \to [x_1, \ldots, x_m]\ (\mathcal{P}; \mathcal{F}) \in P$, there is a rule $A \to [x_1, \ldots, x_m]\ (\lambda, \mu; \delta) \in P'$, with

- $\lambda = \text{cnt}(A) + \text{cnt}(\mathcal{P}) = \text{cnt}(A) + \text{cnt}(\emptyset) = \text{cnt}(A)$,
- $\mu = \text{cnt}(A) + \infty \cdot \text{cnt}(V_{\text{N}} \setminus F)$ and
- $\delta = \text{cnt}([x_1, \ldots, x_m]) - \text{cnt}(A)$. □

The following example provides an RFCPG and the corresponding BCPG.

Example 3.4. Consider the gallery $\mathcal{G}_{\text{permit}}$ again. It can also be generated by an RFCPG, for example, the RFCPG $G_{\text{forbid}} = (V_{\text{N}}, V_{\text{T}}, P, (S, \sigma))$, where



(a) (b)

(c) (d)

Figure 3.7: Some pictures in $\mathcal{G}_{\text{permit}}$.

$$
\begin{aligned}
S &\rightarrow [A, C, C, B] \\
A &\rightarrow [A', C, C, C]\,(\{B\}\,;\emptyset) \;\mid \\
&\quad\; F\,(\{B\}\,;\emptyset) \\
B &\rightarrow [C, C, C, B']\,(\{A'\}\,;\emptyset) \;\mid \\
&\quad\; F\,(\{F\}\,;\emptyset) \\
A' &\rightarrow A\,(\{B'\}\,;\emptyset) \\
B' &\rightarrow B\,(\{A\}\,;\emptyset) \\
F &\rightarrow C \\
C &\rightarrow [b, b, b, b, b, w, w, b, b, w, w, b, b, b, b, b]
\end{aligned}
$$

Figure 3.8: Rules for grammar G_{permit}

$$
\begin{aligned}
S &\rightarrow [A, C, C, B]\left((1,0,0,0,0,0,0), \infty; (-1,1,1,0,0,0,2)\right) \\
A &\rightarrow [A', C, C, C]\left((0,1,1,0,0,0,0), \infty; (0,-1,0,1,0,0,3)\right) \mid \qquad (3.18) \\
&\quad\ F\left((0,1,1,0,0,0,0), \infty; (0,-1,0,0,0,1,0)\right) \qquad (3.19) \\
B &\rightarrow [C, C, C, B']\left((0,0,1,1,0,0,0), \infty; (0,0,-1,0,1,3)\right) \mid \\
&\quad\ F\left((0,0,1,0,0,1,0), \infty; (0,0,-1,0,0,1,0)\right) \\
A' &\rightarrow A\left((0,0,0,1,1,0,0), \infty; (0,1,0,-1,0,0,0)\right) \\
B' &\rightarrow B\left((0,1,0,0,1,0,0), \infty; (0,0,1,0,-1,0,0)\right) \\
F &\rightarrow C\left((0,0,0,0,0,1,0), \infty; (0,0,0,0,0,-1,1)\right) \\
C &\rightarrow [b,b,b,b,b,w,w,b,b,w,w,b,b,b,b,b]\left((0,0,0,0,0,0,1), (\infty); \right. \\
&\qquad \left. (0,0,0,0,0,0,-1)\right)
\end{aligned}
$$

Figure 3.9: Rules for grammar $G_{\text{permit:bag}}$

$V_{\text{N}} = \{S, A, B, A', B', F, C\}$, $V_{\text{T}} = \{w, b\}$, and P is the set of rules in Figure 3.10. The terminal w represents a white circle on a grey background, while the terminal b represents a black circle on the same background.

Let the set of the elements in V_{N} be $\{S, A, B, A', B', F, C\}$. Then the corresponding BCPG is $G_{\text{forbid:bag}} = (V_{\text{N}}, V_{\text{T}}, P', (S, \sigma), [7], (1,0,0,0,0,0,0))$, where P' is the set of rules in Figure 3.11. The bag positions 1 to 7 count the occurrences of S, A, B, A', B', F and C, respectively. Consider Rule 3.20. It can only be applied if the fifth and sixth positions in the lower and upper limits are 0, that is, if there are no occurrences of the variables B' and F in the pictorial form.

3.3.4 The Special Case of CFPGs

In this section, we consider context-free picture grammars. We derive the special case of Lemma 3.1 for this class of grammars where both the permitting and forbidding context sets in every rule are empty. We give an example of grammar that demonstrates how CFPG can be rewritten as BCPG.

Lemma 3.4. Let $G = (V_{\text{N}}, V_{\text{T}}, P, (S, \sigma))$ be a CFPG. Let the set of the elements in V_{N} be arbitrary but fixed. Then there is a BCPG $G' = (V_{\text{N}}, V_{\text{T}}, P', (S, \sigma), I, \beta_0)$ such that $\mathcal{G}(G') = \mathcal{G}(G)$. In G', $\beta_0 = \text{cnt}(S)$. Moreover, for every rule $A \rightarrow [x_1, \ldots, x_m]\ (\mathcal{P}; \mathcal{F}) \in P$, there is a rule $A \rightarrow [x_1, \ldots, x_m]\ (\lambda, \mu; \delta) \in P'$, with

- $\lambda = \text{cnt}(A) + \text{cnt}(\mathcal{P}) = \text{cnt}(A) + \text{cnt}(\emptyset) = \text{cnt}(A)$,

$$
\begin{aligned}
S &\rightarrow [B, C, B, C, A, C, B, C, B] \\
A &\rightarrow [B, C, B, C, A, C, B, C, B]\,(\emptyset; \{B', F\}) \mid \\
&\quad\; F\,(\emptyset; \{A', B'\}) \\
B &\rightarrow [C, C, C, C]\,(\emptyset; \{A, F\}) \mid \\
&\quad\; F\,(\emptyset; \{A, A'\}) \\
A' &\rightarrow A\,(\emptyset; \{B\}) \\
B' &\rightarrow B\,(\emptyset; \{A'\}) \\
F &\rightarrow C\,(\emptyset; \{A, B\}) \\
C &\rightarrow [b, b, b, b, b, w, w, b, b, w, w, b, b, b, b, b]
\end{aligned}
$$

Figure 3.10: Rules for grammar G_{forbid}

$$
\begin{aligned}
S &\rightarrow [A, C, C, B]\,((1,0,0,0,0,0,0), \infty; (1,1,1,0,0,0,2)) \\
A &\rightarrow [A', C, C, C]\,((0,1,0,0,0,0,0), (\infty,\infty,\infty,\infty,0,0,\infty); \\
&\qquad\quad (0,-1,0,1,0,0,3)) \mid \\
&\quad\; F\,((0,1,0,0,0,0,0), (\infty,\infty,\infty,0,0,\infty,\infty); \\
&\qquad\quad (0,-1,0,0,0,1,0)) \\
B &\rightarrow [C, C, C, B']\,((0,0,1,0,0,0,0), (\infty,0,\infty,\infty,\infty,0,\infty); \\
&\qquad\quad (0,0,-1,0,1,0,0,3)) \mid \\
&\quad\; F\,((0,0,1,0,0,0,0), (\infty,0,\infty,0,\infty,\infty,\infty); \\
&\qquad\quad (0,0,-1,0,0,1,0)) \\
A' &\rightarrow A\,((0,0,0,1,0,0,0), (\infty,\infty,0,\infty,\infty,\infty,\infty); \\
&\qquad\quad (0,1,0,-1,0,0,0)) \\
B' &\rightarrow B\,((0,0,0,0,1,0,0), (\infty,\infty,\infty,0,\infty,\infty,\infty); \\
&\qquad\quad (0,0,1,0,-1,0,0)) \\
F &\rightarrow C\,((0,0,0,0,0,1,0), (\infty,0,0,\infty,\infty,\infty,\infty); \\
&\qquad\quad (0,0,0,0,0,-1,1)) \\
C &\rightarrow [b,b,b,b,b,w,w,b,b,w,w,b,b,b,b,b]\,((0,0,0,0,0,0,1), (\infty); \\
&\qquad\quad (0,0,0,0,0,0,-1))
\end{aligned}
\tag{3.20}
$$

Figure 3.11: Rules for grammar $G_{\text{forbid:bag}}$

- $\mu = \text{cnt}(A) + \infty \cdot \text{cnt}(V_\text{N} \setminus \mathcal{F}) = \text{cnt}(A) + \infty \cdot \text{cnt}(V_\text{N} \setminus \emptyset) = \infty$ and
- $\delta = \text{cnt}([x_1, \ldots, x_m]) - \text{cnt}(A)$. □

The following example provides a CFPG and its corresponding BCPG.

Example 3.5. Consider the gallery $\mathcal{G}_\text{free}$. Four pictures in $\mathcal{G}_\text{free}$ are shown in Figure 3.12. The smallest picture (Figure 3.12a) has the following structure: in the centre there are four white circles on a grey background, while the surrounding area is filled with black circles on a grey background. In general, every picture in $\mathcal{G}_\text{free}$ has the following structure: it is sub-divided into an arbitrary number of sub-pictures and a copy of Figure 3.12a placed in each.

$\mathcal{G}_\text{free}$ is generated by the CFPG $G_\text{free} = (V_\text{N}, V_\text{T}, P, (S, \sigma))$, where $V_\text{N} = \{S\}$, $V_\text{T} = \{w, b\}$, and P is the set of rules in Figure 3.13. The terminal w represents a white circle on a grey background, while the terminal b represents a black circle on the same background.

G_free is a simple variation on the grammar in [28, Example 2.1]. The only difference is in the terminals used.

The corresponding BCPG is $G_\text{free:bag} = (V_\text{N}, V_\text{T}, P, (S, \sigma), [1], 1))$, where P is the set of rules in Figure 3.14.

There is only one bag position, it counts S. Consider, for example, Rule 3.21. It can only be applied if the lower limit is 1, that is, if there is at least one occurrence of the variable S in the pictorial form. Clearly, in the case of CFPGs, the bag context adds no control to the underlying grammar.

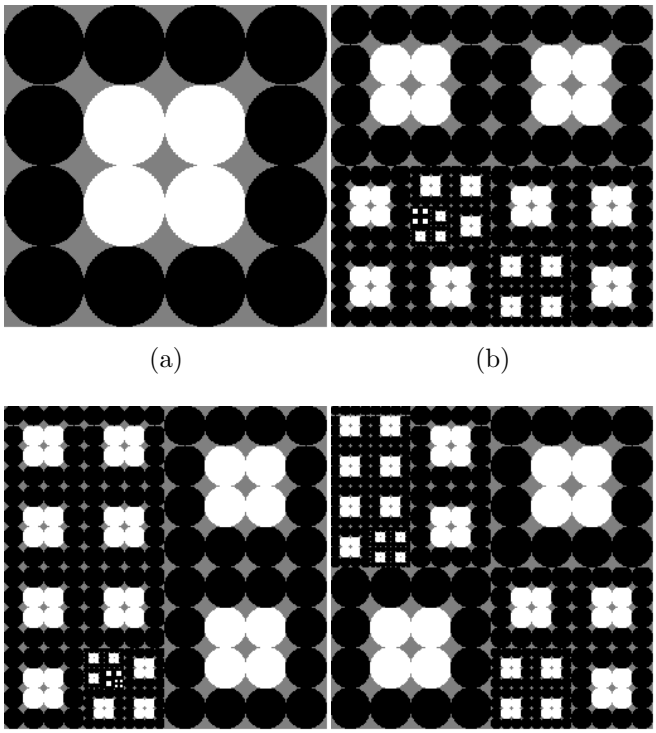
3.3.5 Summary of Rewriting an RCPGs as a BCPGs

In this Section, we demonstrated how to rewrite RCPGs as BCPGs. Rewriting RCPGs as BCPGs is essential for our research as in some instances, we may want to use grammars that have been written in different subclasses of RCPGs. This knowledge helps to adapt many of the existing grammars that have been written using RCPGs.

The next section demonstrates the power of BCPGs over RCPGs in the context of modifying a grammar.

3.4 The Power of Bag Context Picture Grammars

In this section, we consider two galleries, the gallery $\mathcal{G}_\text{random}$ of Example 3.2 and the gallery $\mathcal{G}_\text{permit}$ of Example 3.3. We modify the corresponding BCPG, to create a variation on each gallery. We discuss why it is easier to write a BCPG and modify it again, should we wish to, than an RCPG.



(a) (b)

(c) (d)

Figure 3.12: A selection of pictures in $\mathcal{G}_{\text{free}}$

$$S \to [S,S,S,S] \mid [b,b,b,b,b,w,w,b,b,w,w,b,b,b,b,b]$$

Figure 3.13: Rules for grammar G_{free}

$$S \to [S,S,S,S]\,(1,\infty;3) \mid [b,b,b,b,b,w,w,b,b,w,w,b,b,b,b,b]\,(1,\infty;-1) \tag{3.21}$$

Figure 3.14: Rules for grammar $G_{\text{free:bag}}$

In the first example, we modify the RCPG G_{random} and the BCPG$G_{\text{random:bag}}$ on page 56, to create a variation on the gallery, called $\mathcal{G}_{\text{random:var}}$. We discuss why it is easier to write $G_{\text{random:bag:var}}$ and modify it again, should we wish to, than writing and modifying the RCPG $G_{\text{random:var}}$.

In the variation we show how to use bag context and random context to achieve two objectives in a gallery, namely

1. to regulate the sub-division of the initial square so that there is a lower limit on the size of the smallest sub-square in any picture, and

2. to regulate the distribution of colours in every picture in the gallery.

In this example, the initial square can be sub-divided maximally three times. Moreover, we use three colours (white, purple and green) and represent these as circles. The distribution of colours is such that each picture has $1 + 8 + 8^2 + \ldots + 8^i, i \geq 1$ white circles, 5×8^i purple circles and 3×8^i green circles. Naturally, there are many variations on both aspects of the regulation.

We then discuss how one could achieve the above objectives with random context.

Example 3.6. Consider $\mathcal{G}_{\text{random}}$ of Example 3.2. We modify the grammar $G_{\text{random:bag}}$ to obtain $G_{\text{random:bag:var}}$, which generates the gallery $\mathcal{G}_{\text{random:var}}$. Some pictures in $\mathcal{G}_{\text{random:var}}$ are shown in Figure 3.15.

Like $\mathcal{G}_{\text{random}}$, the gallery $\mathcal{G}_{\text{random:var}}$ consists of a variation on the sequence of pictures approximating the Sierpiński carpet. Each of the smallest pictures in $\mathcal{G}_{\text{random:var}}$ is divided into nine equal-sized sub-squares. The sub-square in the centre is filled with a white circle on a grey background, while the remaining sub-squares are filled with five purple and three green circles on a grey background. Figure 3.15a is one example of a smallest picture; the other smallest pictures differ from Figure 3.15a in the placement of the purple and green circles.

Each of the second smallest pictures in $\mathcal{G}_{\text{random:var}}$ has the following structure: the white circle is as in Figure 3.15a, while each of the remaining eight sub-squares is sub-divided into nine equal-sized sub-squares like Figure 3.15a. Once the sub-division is complete, the 72 sub-squares are coloured such that there are $5 \times 8 = 40$ purple and $3 \times 8 = 24$ green circles. Figure 3.15b is one example of a second smallest picture; the other second smallest pictures differ from Figure 3.15b in the placement of the non-white circles.

Each of the third smallest pictures, for example, Figure 3.15c, has the following structure: the white circles are as in Figure 3.15b, while each remaining square is sub-divided into nine equal-sized sub-squares like

Figure 3.15a. Once the sub-division is completed, the sub-squares are coloured such that there are 5×8^2 purple and 3×8^2 green circles.

This pattern continues from one picture in the sequence to the next.

$\mathcal{G}_{\text{random:var}}$ is generated by the BCPG $G_{\text{random:bag:var}} = (V_{\text{N}}, V_{\text{T}}, P', (S, \sigma), [9], (1, 0, 0, 0, 0, 0, 0, 0, 0))$ where $V_{\text{N}} = \{S, T, U, F, C\}$, $V_{\text{T}} = \{b, w, g\}$ and P' is the set of rules in Figure 3.16. Terminals w, b and g represent white, purple and green circles, respectively.

$\mathcal{G}_{\text{random:var}}$ can also be generated by the RCPG $G_{\text{random:var}} = (V_{\text{N}}, V_{\text{T}}, P, (S, \sigma))$ where $V_{\text{N}} = \{S, S', S'', T, T', T'', U, U', F, B, B', B'', B''', B'''', D, D', D'', C\}$, $V_{\text{T}} = \{b, w, g\}$ and P is the set of rules in Figure 3.17. Terminals w, b and g represent white, purple and green circles, respectively.

Please note that we have added one variable (C), one terminal (g), several rules and four bag positions to $G_{\text{random:bag}}$ in order to create $G_{\text{random:bag:var}}$.

As in $G_{\text{random:bag}}$, the bag positions 1 to 4 in $G_{\text{random:bag:var}}$ count the occurrences of the variables S, T, U and F, respectively. Positions 5 and 6 count the number of terminals b and g, respectively. These terminals must be counted with respect to modulus 5, respectively 3, to ensure that, for every white circle on the lowest level of refinement, there are five purple and three green circles. The last bag position counts how many times Rule 3.23 has been applied. This allows us to place an upper limit on the number of times the initial square is sub-divided and, therefore, place a lower limit on the size of the smallest circle in any picture.

Rule 3.22 divides every square labelled S into nine equal-sized sub-squares, eight of which are labelled T and the central one w. All occurrences of T can turn into U (Rule 3.23) and then S again (Rule 3.24). Therefore the initial square is divided into increasingly smaller sub-squares. All sub-squares are of the same size, apart from those that are labelled by the terminal w. However, the cycle of rules, Rules 3.22–3.23–3.24–3.22... cannot be repeated arbitrarily often, unlike the corresponding rules in $G_{\text{random:bag}}$. In $G_{\text{random:bag:var}}$, Rule 3.23 can be applied maximally 72 times, as bag position 7 of the upper limit is set to 71. Therefore the sub-squares cannot become arbitrarily small.

Once this cycle has stopped, every label T is eventually turned into C, which becomes one of b, g or C in a specific order. Consider Rules 3.25, 3.26 and 3.27. Rule 3.25 has to be applied exactly five times before Rule 3.26 can be applied. Similarly, Rule 3.26 has to be applied exactly three times before Rule 3.27 can be applied. The latter rule resets the counters for terminals b and g to zero. Once Rule 3.27 has been applied exactly once, Rule 3.25 is enabled again. This cycle is enforced by positions 5 and 6 of the lower and upper limits in these rules. This ensures that, for every white circle on the lowest level of refinement, there are five purple and three green circles.

We now discuss how each of the above two objectives could be achieved

with random context.

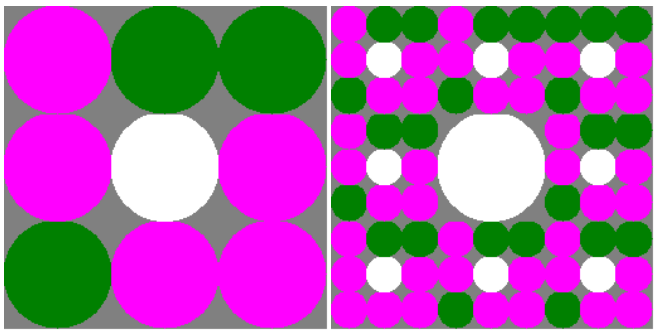
Objective 1: Consider the cycle of rules, Rules 3.22–3.23–3.24–3.22... again. This cycle sub-divides the initial square into increasingly smaller sub-squares. The initial square can be sub-divided maximally three times. This is enforced by setting bag position 7 of the upper limit of Rule 3.23 to 71. In an RCPG, such an upper limit can be achieved by using so-called coloured versions of a variable. Consider the RCPG G_{random} in Figure 3.2 again. In G_{random} the initial square may not be sub-divided more than three times. This upper limit can be achieved by using coloured variables $S, S', S'', T, T', T'', U, U'$, and coloured versions of Rules 3.9, 3.10, 3.11, 3.12 and 3.13 of G_{random}. This is shown in Figure 3.17.

In an RCPG it is, therefore, possible to ensure that the initial square is not sub-divided more than a predetermined number of times, say n. However, in general, the RCPG requires more variables and rules than the corresponding BCPG. Moreover, every time we change n, say from 3 to 5, because, for example, the initial square is on a bigger screen, the number of variables and rules in the RCPG increases whereas, in the corresponding BCPG, only one component of an upper limit needs to be changed, in this case, bag position 7 of the upper limit of Rule 3.23.

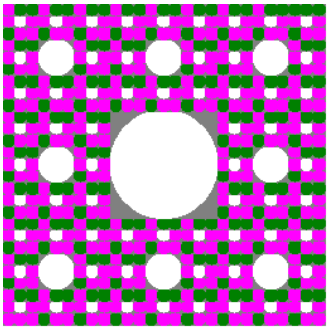
Regarding Objective 2: Consider Rules 3.25-3.27 again. These rules ensure that the first five squares become purple and three squares green. This is enforced by positions 5 and 6 of the lower and upper limits of Rules 3.25 and 3.26.

In an RCPG such a colour distribution can be achieved by using coloured variables. Consider the RCPG G_{random} in Figure 3.2 again. For simplicity, assume that Ts are generated in multiples of five. Of these, three must become purple and two green. This can be achieved by using coloured variables $B, B', B'', B''', B'''', D, D', D''$, and coloured versions of Rules 3.11, 3.12 and 3.14 of G_{random}. This is shown in Figure 3.17. Intuitively, one C after another is turned into $B, B', B'', B''', B'''', D, D'$ and G'', in this order. Then variables B, B', B'', B''' and B'''' are turned into b, and subsequently D, D' and D'' into g. The random context ensures that the rules are applied in the order in which they are listed.

In an RCPG it is, therefore, possible to ensure that colours are allocated in a predetermined ratio. However, in general, the RCPG requires more variables and rules than the corresponding BCPG. Moreover, every time we change the colour distribution, say from three purple and two green to four purple and three green, the number of variables and rules in the RCPG increases and the rules themselves may have to change, in particular the random context. In the corresponding BCPG, only selected components of the lower and upper limits of some rules need to be changed, in this case, positions 5 and 6 of the lower and upper limits of Rules 3.25-3.27.



(a) (b)



(c)

Figure 3.15: Some pictures in $\mathcal{G}_{\text{random:var}}$

$$
\begin{aligned}
S \rightarrow\ & [T,T,T,T,w,T,T,T,T]\,((1,0,0,0,0,0,0) \\
& (\infty,\infty,0,\infty,\infty,\infty,\infty);(-1,8,0,0,0,0,0)) && (3.22)\\
T \rightarrow\ & U\,((0,1,0,0,0,0,0)\,,(0,\infty,\infty,0,\infty,\infty,71)\,; \\
& \qquad (0,-1,1,0,0,0,0))\ \mid && (3.23)\\
& F\,((0,1,0,0,0,0,0)\,,(0,\infty,0,0,\infty,\infty,\infty)\,; \\
& \qquad (0,-1,0,1,0,0,0))\ \mid \\
& C\,((0,1,0,1,0,0,0)\,,\infty;(0,-1,0,0,0,0,0)) \\
U \rightarrow\ & S\,((0,0,1,0,0,0,0)\,,(\infty,0,\infty,\infty,\infty,\infty,\infty)\,; \\
& \qquad (1,0,-1,0,0,0,0)) && (3.24)\\
F \rightarrow\ & C\,((0,0,0,1,0,0,0)\,,(\infty,0,\infty,\infty,\infty,\infty,\infty)\,; \\
& \qquad (0,0,0,-1,0,0,0)) \\
C \rightarrow\ & b\,((0,0,0,0,0,0,0)\,,(\infty,\infty,\infty,\infty,4,\infty,\infty)\,; \\
& \qquad (0,0,0,0,1,0,0))\ \mid && (3.25)\\
& g\,((0,0,0,0,5,0,0)\,,(\infty,\infty,\infty,\infty,\infty,2,\infty)\,; \\
& \qquad (0,0,0,0,0,1,0))\ \mid && (3.26)\\
& C\,((0,0,0,0,5,3,0)\,,\infty;(0,0,0,0,-5,-3,0)) && (3.27)
\end{aligned}
$$

Figure 3.16: Rules for grammar $G_{\text{random:bag:var}}$

$$
\begin{aligned}
S &\to [T,T,T,T,w,T,T,T,T] \\
T &\to U\,(\emptyset;\{S,T''\}) \mid \\
&\quad\ T''\,(\emptyset;\{U\}) \\
U &\to S'\,(\emptyset;\{T\}) \\
S' &\to [T',T',T',T',w,T',T',T',T']\,(\emptyset;\{U\}) \\
T' &\to U'\,(\emptyset;\{S',T''\}) \mid \\
&\quad\ T''\,(\emptyset;\{S',U\}) \\
U' &\to S''\,(\emptyset;\{T'\}) \\
S'' &\to [T'',T'',T'',T'',w,T'',T'',T'',T'']\,(\emptyset;\{U\}) \\
T'' &\to F\,(\emptyset;\{S,S',S'',U,U',F\}) \mid \\
&\quad\ C\,(\{F\};\{B,B',B'',B''',B'''',D,D',D'',T,T',S',S''\}) \\
F &\to C\,(\emptyset;\{T''\}) \\
C &\to B\,(\emptyset;\{B,T'',D''\}) \mid \\
&\quad\ B'\,(\{B\};\{B'\}) \mid \\
&\quad\ B''\,(\{B'\};\{B''\}) \mid \\
&\quad\ B'''\,(\{B''\};\{B'''\}) \mid \\
&\quad\ B''''\,(\{B'''\};\{B''''\}) \mid \\
&\quad\ D\,(\{B''''\};\{D\}) \mid \\
&\quad\ D'\,(\{D\};\{D'\}) \mid \\
&\quad\ D''\,(\{D'\};\{D''\}) \\
B &\to b\,(\{D''\};\{T''\}) \\
B' &\to b\,(\emptyset;\{B\}) \\
B'' &\to b\,(\emptyset;\{B,B'\}) \\
B''' &\to b\,(\emptyset;\{B,B',B''\}) \\
B'''' &\to b\,(\emptyset;\{B,B',B'',B'''\}) \\
D &\to g\,(\emptyset;\{B,B',B'',B''',B''''\}) \\
D' &\to g\,(\emptyset;\{B,B',B'',B''',B'''',D\}) \\
D'' &\to g\,(\emptyset;\{B,B',B'',B''',B'''',D,D'\})
\end{aligned}
$$

Figure 3.17: Coloured rules for grammar $G_{\text{random:var}}$

In the Examples 3.7 and 3.8, we modify the BCPGs, $G_{\text{permit:bag}}$ and G_{permit}, to create variations on the gallery $\mathcal{G}_{\text{permit}}$. We discuss why it is easier to write and modify a BCPG again, should we wish to, than writing and modifying an RCPG.

Example 3.7. In this example, we modify grammar $G_{\text{permit:bag}}$ to obtain $G_{\text{permit:bag:varA}}$, which generates the gallery $\mathcal{G}_{\text{permit:varA}}$. Four pictures in $\mathcal{G}_{\text{permit:varA}}$ are shown in Figure 3.18, where Figure 3.18a is the smallest picture in this gallery, Figure 3.18b the second smallest, and so on.

Let there be an image called π. The image π has the following structure: in the centre there are four white circles on a grey background, while the surrounding area is filled with black circles on a grey background.

In $\mathcal{G}_{\text{permit:varA}}$, every picture in the gallery has the following structure: the top left and bottom right quarter of a picture each consist of π. The bottom left quarter of the picture is divided into four, and π is placed in each quarter except the bottom right quarter which is divided into four again and π placed in each quarter. This may be recursively repeated for the bottom left quarter of the resulting sub-picture. The top right quarter must be divided into four and π placed in each quarter every time the bottom left quarter is subdivided. Hence the bottom left quarter of the picture is always sub-divided two more times than the top right quarter of the picture.

The terminal w represents a white circle on a grey background, while the terminal b represents a black circle on the same background.

Let the set of the elements in $V_{\text{N}} = \{S, A, B, A', B', F, C\}$. Then $\mathcal{G}_{\text{permit:varA}}$ is generated by BCPG $G_{\text{permit:bag:varA}} = (V_{\text{N}}, V_{\text{T}}, P', (S, \sigma), [6], (1, 0, 0, 0, 0, 0))$, where P' is the set of rules in Figure 3.19.

The corresponding RCPG $G_{\text{permit:varA}} = (V_{\text{N}}, V_{\text{T}}, P, (S, \sigma))$, where $V_{\text{N}} = \{S, A, B, A', B', A'', A''', A'''', F, C\}$, $V_{\text{T}} = \{w, b\}$, and P is the set of rules in Figure 3.20.

In $G_{\text{permit:bag:varA}}$ the bag positions 1 to 5 count the occurrences of the variables A, B, A', B' and F respectively. Bag position 6 counts the number of times Rule 3.28 has been applied. This allows us to place an upper limit on the number of times the bottom right quarter is sub-divided before any other rule can be applied.

It is also possible in an RCPG to ensure that one quarter is sub-divided a number of times, say n, before another quarter can be sub-divided. However, the RCPG requires more variables and rules than the corresponding BCPG. $G_{\text{permit:varA}}$ has one more variable and rule than $G_{\text{permit:bag:varA}}$.

Example 3.8. In this example, we modify grammar $G_{\text{permit:bag}}$ again to obtain $G_{\text{permit:bag:varB}}$, which generates the gallery $\mathcal{G}_{\text{permit:varB}}$. Two pictures in $\mathcal{G}_{\text{permit:varB}}$ are shown in Figure 3.21, where Figure 3.21a is the smallest picture in this gallery and Figure 3.21b the second smallest.

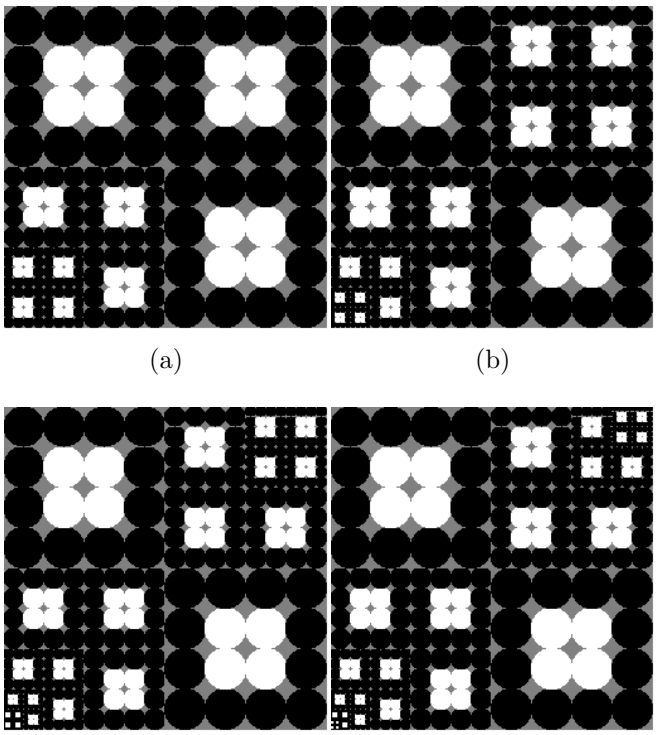


Figure 3.18: Some pictures in $\mathcal{G}_{\text{permit:varA}}$

$$
\begin{aligned}
S &\rightarrow [A,C,C,B]\left((0,0,0,0,0,0),\infty;(1,1,0,0,0,0)\right) \\
A &\rightarrow [A',C,C,C]\left((1,0,0,0,0,0),(\infty,\infty,\infty,0,0,1);\right. \\
&\qquad \left.(-1,0,1,0,0,1)\right) \mid && (3.28) \\
&\quad F\left((1,0,0,0,0,2),(\infty,\infty,0,0,\infty,\infty);(-1,0,0,0,1,0)\right) && (3.29) \\
B &\rightarrow [C,C,C,B']\left((0,1,0,0,0,2),(\infty,\infty,\infty,\infty,0,\infty);\right. \\
&\qquad \left.(0,-1,0,1,0,-1)\right) \mid && (3.30) \\
&\quad F\left((0,1,0,0,0,0),(0,\infty,0,\infty,\infty,\infty);(1,-1,0,0,0,1)\right) \\
A' &\rightarrow A\left((0,0,1,0,0,0),\infty;(1,0,-1,0,0,0)\right) \\
B' &\rightarrow B\left((0,0,0,1,0,0),(\infty,\infty,0,\infty,\infty,\infty);(0,1,0,-1,0,0)\right) \\
F &\rightarrow C\left((0,0,0,0,1,0),(0,0,\infty,\infty,\infty,\infty);(0,0,0,0,-1,0)\right) \\
C &\rightarrow [b,b,b,b,b,w,w,b,b,w,w,b,b,b,b,b]
\end{aligned}
$$

Figure 3.19: Set of rules for grammar $G_{\text{permit:bag:varA}}$

$$
\begin{aligned}
S &\rightarrow [A, C, C, B] \\
A &\rightarrow [A', C, C, C]\,(\emptyset; \{B', F\}) \\
A' &\rightarrow [A'', C, C, C]\,(\emptyset; \{B', F\}) \\
B &\rightarrow [C, C, C, B']\,(\{A''\}; \{F\}) \mid \\
&\quad\; F\,(\{F\}; \emptyset) \\
A'' &\rightarrow F\,(\emptyset; \{B', F\}) \mid \\
&\quad\; A'\,(\{B'\}; \emptyset) \\
B' &\rightarrow B\,(\{A'\}; \{F\}) \\
F &\rightarrow C\,(\emptyset; \{B\}) \\
C &\rightarrow [b, b, b, b, b, w, w, b, b, w, w, b, b, b, b, b]
\end{aligned}
$$

Figure 3.20: Set of rules for grammar $G_{\text{permit:varA}}$

Let there be an image called π. The image π has the following structure: in the centre there are four white circles on a grey background, while the surrounding area is filled with black circles on a grey background.

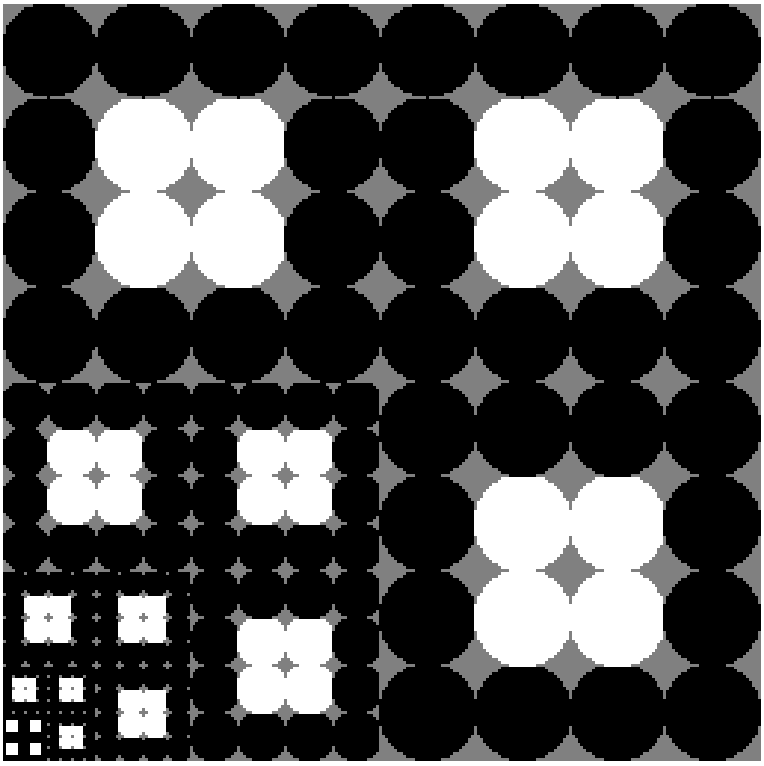
In $\mathcal{G}_{\text{permit:varB}}$, every picture in the gallery has the following structure: the top left and bottom right quarter of a picture each consist of π. The bottom left quarter of the picture is divided into four, and π is placed in each quarter except the bottom right quarter which is divided into four again and π placed in each quarter except the bottom right quarter. The subdivision of the bottom right quarter is repeated twice more and π placed in each quarter. This may be recursively repeated for the bottom left quarter of the resulting sub-picture. The top right quarter must be divided into four and π placed in each quarter every time the bottom left quarter is subdivided. Hence the bottom left quarter of the picture is always sub-divided four times more than the top right quarter of the picture.

The terminal w represents a white circle on a grey background, while the terminal b represents a black circle on the same background.

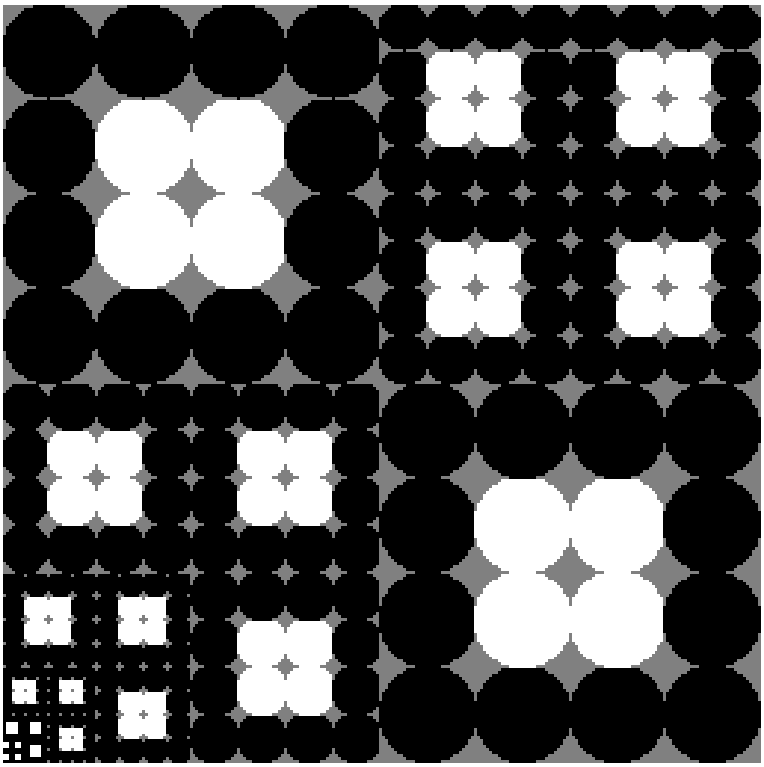
Let the set of the elements in $V_{\text{N}} = \{S, A, B, A', B', F, C\}$, then $\mathcal{G}_{\text{permit:varB}}$ is generated by BCPG $G_{\text{permit:bag:varB}} = (V_{\text{N}}, V_{\text{T}}, P', (S, \sigma), [6], (1, 0, 0, 0, 0, 0))$, where P' is the set of rules in Figure 3.22

The corresponding RCPG $G_{\text{permit:varB}} = (V_{\text{N}}, V_{\text{T}}, P, (S, \sigma))$, where $V_{\text{N}} = \{S, A, B, A', B', A'', F, C\}$, $V_{\text{T}} = \{w, b\}$, and P is the set of rules in Figure 3.23.

As in $G_{\text{permit:bag:varA}}$, in $G_{\text{permit:bag:varB}}$ the bag positions 1 to 5 count the occurrences of the variables A, B, A', B' and F respectively. Bag position 6 counts the number of times Rule 3.28 has been applied. This allows us to place an upper limit on the number of times the bottom right quarter is



(a)



(b)

Figure 3.21: Some pictures in $\mathcal{G}_{\text{permit:varB}}$

sub-divided before any other rule can be applied. Both $G_{\text{permit:bag:varA}}$ and $G_{\text{permit:bag:varB}}$ have the same number of rules and bag positions, they only differ in some values of the lower and upper limit of some rules as follows:

- Rule 3.31, where the last position of the upper limit was changed to 3 from 1 in Rule 3.28,
- the value of the last position of the lower limit of Rule 3.32 was set to 4 from 2 in Rule 3.29,
- lastly, the value of the last position of the lower limit of Rule 3.33 was set to 4 from 2 in Rule 3.30.

This gallery can also be generated by an RCPG but it requires more variables and rules than the corresponding BCPG. $G_{\text{permit:varB}}$ has three more variables and rules than $G_{\text{permit:bag:varB}}$.

Hence it is possible in an RCPG to ensure that sub-pictures are sub-divided in defined intervals. However, we have observed that the RCPG requires more variables and rules than the corresponding BCPG. Furthermore, every time we change the number of intervals for sub-division, for example, from two to four in this case, the number of variables and rules in the RCPG increased and some rules changed. On the other hand, in the corresponding BCPG only a few values of the lower and upper limits of the rules changed.

$$
\begin{aligned}
S &\rightarrow [A,C,C,B]\left((0,0,0,0,0,0),\infty;(1,1,0,0,0,0)\right) \\
A &\rightarrow [A',C,C,C]\left((1,0,0,0,0,0),(\infty,\infty,\infty,0,0,3);\right. \\
&\qquad \left.(-1,0,1,0,0,1)\right) \mid && (3.31) \\
&\quad F\left((1,0,0,0,0,4),(\infty,\infty,0,0,\infty,\infty);(-1,0,0,0,1,0)\right) && (3.32) \\
B &\rightarrow [C,C,C,B']\left((0,1,0,0,0,4),(\infty,\infty,\infty,\infty,0,\infty);\right. \\
&\qquad \left.(0,-1,0,1,0,-1)\right) \mid && (3.33) \\
&\quad F\left((0,1,0,0,0,0),(0,\infty,0,\infty,\infty,\infty);(1,-1,0,0,0,1)\right) \\
A' &\rightarrow A\left((0,0,1,0,0,0),\infty;(1,0,-1,0,0,0)\right) \\
B' &\rightarrow B\left((0,0,0,1,0,0),(\infty,\infty,0,\infty,\infty,\infty);(0,1,0,-1,0,0)\right) \\
F &\rightarrow C\left((0,0,0,0,1,0),(0,0,\infty,\infty,\infty,\infty);(0,0,0,0,-1,0)\right) \\
C &\rightarrow [b,b,b,b,b,w,w,b,b,w,w,b,b,b,b,b] && (3.34)
\end{aligned}
$$

Figure 3.22: Set of rules for grammar $G_{\text{permit:bag:varB}}$

$$
\begin{aligned}
S &\rightarrow [A, C, C, B] \\
A &\rightarrow [A', C, C, C]\,(\emptyset; \{B', F\}) \\
A' &\rightarrow [A'', C, C, C]\,(\emptyset; \{B', F\}) \\
A'' &\rightarrow [A''', C, C, C]\,(\emptyset; \{B', F\})) \\
A''' &\rightarrow [A'''', C, C, C]\,(\emptyset; \{B', F\}) \\
B &\rightarrow [C, C, C, B']\,(\{A''\}; \{F\}) \;\mid \\
&\quad\; F\,(\{F\}; \emptyset) \\
A'''' &\rightarrow F\,(\emptyset; \{B', F\})) \;\mid \\
&\quad\; A'\,(\{B'\}; \emptyset) \\
B' &\rightarrow B\,(\{A'\}; \{F\}) \\
F &\rightarrow C\,(\emptyset; \{B\}) \\
C &\rightarrow [b, b, b, b, b, w, w, b, b, w, w, b, b, b, b, b]
\end{aligned}
$$

Figure 3.23: Set of rules for grammar $G_{\text{permit:varB}}$

3.5 Conclusion

In this chapter, we introduced bag context picture grammars. We stated a lemma that shows how any RCPG can be written as a BCPG. We considered this process for three sub-classes of RCPGs and, in each case, illustrated it with an example. The knowledge of how to rewrite RCPGs as BCPGs is essential for our research as it helps to adapt many of the existing grammars that have been written using RCPGs.

We then considered three sets of pictures and discussed why it is easier to write a BCPG that generates a variation on the set, and modify that BCPG again, should we wish to, than to do the same to the RCPG that generates the original picture set. One reason for this difference is that, in random context, it is not possible to determine how many of the permitting or forbidding variables are present in the developing picture, since only the presence or absence of a variable matters whereas, in bag context, it is possible to keep track of the number of variables in the bag. Another reason is that, in bag context, the application of a rule does not necessarily depend on the presence or absence of variables. Other conditions for a rule to be applicable may be defined, for example, the number of times this rule or another has already been applied. This information can also be kept in the bag. This makes it possible, for example, to avoid having parts of a picture that are too small for the human eye to recognise, for example, limiting the number of refinements of the produced pictures.

BCPGs like RCPGs have shown to be capable of generating pictures with

variations which is vital in the generation of structurally similar pictures. Moreover, in the generation of structurally similar pictures, we often require the level of refinement to be between predetermined bounds. BCPGs require fewer rules and variables to do this than RCPGs. Furthermore, when the bounds are changed, in a BCPG few changes to the grammar are required, often only the values of the bag, while in an RCPG the grammar has to be rewritten. Hence the definition of BCPGs has provided a grammar type which is simple to use in the framework of regulating the application of rules.

In the next chapter, we discuss the generation of similar pictures using BCPGs and RCPGs. We used both BCPGs and RCPGs to compare which grammar type performs better in the generation of structurally similar pictures.

Chapter 4

Generation of Similar Pictures using BCPGs and RCPGs

4.1 Introduction

This chapter discusses methods of generating structurally similar pictures using bag context picture grammars and random context picture grammars. Some parts of this chapter have been submitted at Advances in Intelligent Systems and Computing Series book (Springer) [44].

In the previous chapter, we have introduced BCPG. We showed that any RCPG can be written as a BCPG and illustrated it with examples. Moreover, We considered two sets of pictures and discussed why it was easier to write BCPGs that generate a variation on the sets and modify those BCPGs again, should we wish to, than to do the same to the RCPGs that generate the original picture sets. Both RCPGs and BCPGs can produce pictures with variations, but we have observed that BCPGs were easier to write and rewrite if we wished to. In this chapter, both BCPGs and RCPGs were used to develop methods of generating structurally similar pictures.

This research of generation in a controlled manner of a finite number of similar pictures for visual password systems was motivated by Okundaye et al. [64]. Generation of similar pictures is important for applications that need the use of similar pictures such as visual password systems. The problem with many picture grammars today is that they have no way of generating structurally similar pictures together and if they can, they require a lot of rules and variables to do so. Example a picture grammar that generates the same picture at a particular level of refinement. Many picture grammars can generate similar pictures, but they fail to generate only the relevant similar pictures. Okundaye et al. [63] generated similar pictures using various grammars which generated infinite number of pictures for visual password authentication. Furthermore, in their research, they presented pictures on consecutive refinement levels in the same grammar but failed to generate

galleries that contain only structurally similar pictures because some pictures in the galleries were quite different to the query picture. It is difficult to tell the difference in some of the pictures presented as the difference was too tiny for the human eye to detect it, while in other pictures the gap was too high for the pictures to be considered similar [63, 64].

This chapter presents methods developed in generating structurally similar pictures using BCPGs and RCPGs for use in visual password systems. The generated pictures are within some defined bounds with a small gap in the levels of refinement. Refinement level is the number of times a picture is divided to create sub-pictures. The difference between one picture and another is either in the placement of coloured objects or the refined part rather than the level of refinement.

Moreover, we investigate which grammar type among RCPGs and BCPGs is the easiest to use in generating galleries of structurally similar pictures. We recommend a grammar type that is easy to modify, easy to track changes and that does not require significant changes to the grammar. Ideally, the grammar should require minimal introduction or omission of variables and rules for each modification. Its structure, that is rules and context, should require few changes when modified. It should be easy to identify changes and, for further similar modifications, it should be easy to identify where to change.

4.2 Generation of Similar Pictures

We started our evaluation with a BCPG and we wrote its corresponding RCPG. We then selected one picture generated by the grammar, identified some of its characteristics and modified the grammar to generate only pictures that have some characteristics of the selected picture, hence creating a gallery of structurally similar pictures. The modified grammar generates a gallery that is a subset of the initial gallery. The second step may be repeated to create another gallery as desired.

4.2.1 Limiting the Refinement Levels

In this section, we generate structurally similar pictures using BCPGs and RCPGs with the main feature being the level of refinement. For each example, we select a picture at a particular level of refinement and generate a gallery of pictures that are only at that level of refinement. The colouring of objects in the generated pictures varies, creating pictures that are structurally similar regarding their size but different concerning the placement of colours. Four examples will be provided for this section.

The following example provides a BCPG and an RCPG that generate pictures at first, second and third levels of refinement.

Example 4.1. Consider the gallery $\mathcal{G}_{\text{circlesMixed}}$ which consists of a sequence of pictures approximating the Sierpiński carpet. Some pictures in $\mathcal{G}_{\text{circlesMixed}}$ are shown in Figure 4.1. The first refinement pictures in $\mathcal{G}_{\text{circlesMixed}}$ (see Figures 4.1a and 4.1e for example), are divided into nine equal-sized sub-squares. The middle sub-square is filled with a white circle on a grey background. The remaining eight sub-squares are filled with five light grey circles and three black circles on a grey background in no particular order.

The second refinement pictures in $\mathcal{G}_{\text{circlesMixed}}$ (examples shown in Figure 4.1b and Figure 4.1d) have the following format: There is a white circle at the centre as in Figure 4.1a, while each of the remaining eight sub-squares is sub-divided into nine equal-sized sub-squares like Figure 4.1a. After the second sub-division, the remaining sub-squares are coloured in such a way that for every five light grey circles there are three black circles. The non-white circles are placed in no particular order.

Furthermore, the third refinement pictures in $\mathcal{G}_{\text{circlesMixed}}$ (examples are shown in Figures 4.1f and 4.1c) have the following format: the white circles are as in Figure 4.1d, while each of the remaining eight sub-squares is sub-divided into nine equal-sized sub-squares like Figure 4.1a. After sub-division, the remaining sub-squares are coloured in such a way that, for every five light grey circles, there are three black circles. The order of the non-white circles will be different every picture at this refinement level.

$\mathcal{G}_{\text{circlesMixed}}$ is generated by the BCPG $G_{\text{circlesMixed:bag}} = (V_{\text{N}}, V_{\text{T}}, P', (S, \sigma), [8], (1, 0, 0, 0, 0, 0, 0, 0))$ where $V_{\text{N}} = \{S, T, U, F, C\}$, $V_{\text{T}} = \{b, w, g\}$ and P' is the set of rules in Figure 4.2. Terminals w, b and g represent white, purple and green circles, respectively.

The bag positions 1 to 5 in $G_{\text{circlesMixed:bag}}$ count the occurrences of S, T, U, F and C respectively. Bag positions 6 and 7 count the occurrences of b and g respectively. We count these terminals to ensure that, for every white circle on the lowest level of refinement, there are five light grey and three black circles. The last position counts how many times Rule 4.1 in Figure 4.2 has been applied. This helps us put a lower and upper limit on the subdivision of the initial square.

The gallery $\mathcal{G}_{\text{circlesMixed}}$ can also be generated by RCPG $G_{\text{circlesMixed}} = (V_{\text{N}}, V_{\text{T}}, P, (S, \sigma))$ where $V_{\text{N}} = \{S, S', S'', T,\ T',\ T'', U, U', F, C, B, B',\ B'',\ B''', B'''',\ D, D', D''\}$, $V_{\text{T}} = \{w,\ b,\ g\}$ and P is the set of rules in Figure 4.3. Terminals w, b and g represent white, purple and green circles, respectively.

The RCPG $G_{\text{circlesMixed}}$ requires more variables and rules to generate gallery $\mathcal{G}_{\text{circlesMixed}}$ than its corresponding BCPG $G_{\text{circlesMixed:bag}}$. Additional variables S', S'', T', T'' and U' were added to ensure that the initial

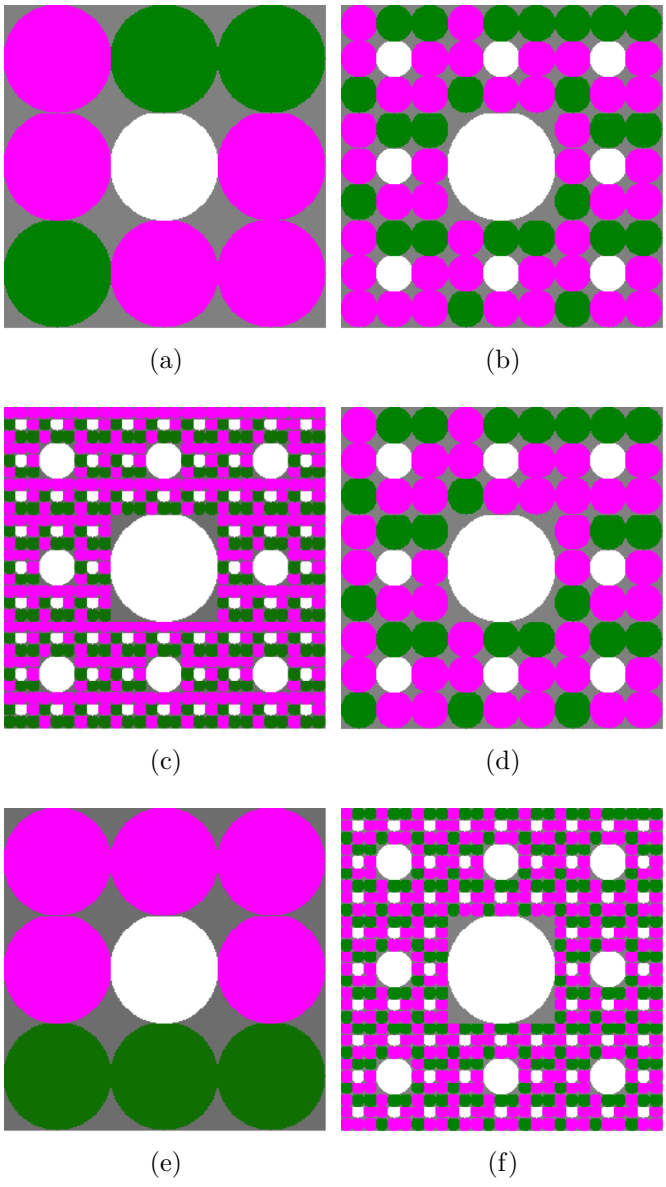


Figure 4.1: A selection of pictures in $\mathcal{G}_{\text{circlesMixed}}$

square is not divided more than three times. We have also added variables B, B', B'', B''', B'''', D, D' and D'' to ensure that for every five light grey circles there are three black circles. Furthermore, there are nineteen more rules in the RCPG $G_{\text{circlesMixed}}$ than in its corresponding BCPG $G_{\text{circlesMixed:bag}}$. The next example generates only pictures that are in the first refinement.

$$
\begin{aligned}
S \rightarrow\ & [T,T,T,T,w,T,T,T,T]\,((1,0,0,0,0,0,0,0,0),\\
& (\infty,\infty,0,\infty,\infty,\infty,\infty,\infty);(-1,8,0,0,0,0,0,0,0))\\
T \rightarrow\ & U\,((0,1,0,0,0,0,0,0,0)\,,(0,\infty,\infty,0,\infty,\infty,\infty,71)\,;\\
& \qquad (0,-1,1,0,0,0,0,1))\ \mid && (4.1)\\
& F\,((0,1,0,0,0,0,0,0,0)\,,(0,\infty,0,0,\infty,\infty,\infty,0)\,; && (4.2)\\
& \qquad (0,-1,0,1,0,0,0,0))\ \mid && (4.3)\\
& C\,((0,1,0,1,0,0,0,0,0)\,,\infty;(0,-1,0,0,1,0,0,0))\\
U \rightarrow\ & S\,((0,0,1,0,0,0,0,0,0)\,,(\infty,0,\infty,\infty,\infty,\infty,\infty,\infty)\,;\\
& \qquad (1,0,-1,0,0,0,0,0))\\
F \rightarrow\ & C\,((0,0,0,1,0,0,0,0,0)\,,(\infty,0,\infty,\infty,\infty,\infty,\infty,\infty)\,;\\
& \qquad (0,0,0,-1,1,0,0,0))\\
C \rightarrow\ & b\,((0,0,0,0,1,0,0,0)\,,(\infty,\infty,\infty,\infty,\infty,4,\infty,\infty)\,;\\
& \qquad (0,0,0,0,-1,1,0,0))\ \mid\\
& g\,((0,0,0,0,1,5,0,0)\,,(\infty,\infty,\infty,\infty,\infty,\infty,2,\infty)\,;\\
& \qquad (0,0,0,0,-1,0,1,0))\ \mid\\
& C\,((0,0,0,0,1,5,3,0)\,,\infty;(0,0,0,0,0,-5,-3,0))
\end{aligned}
$$

Figure 4.2: Rules for grammar $G_{\text{circlesMixed:bag}}$

Example 4.2. Consider the gallery $\mathcal{G}_{\text{circlesFirst}}$ which consists of structurally similar pictures to Figure 4.1a. This gallery has only the first refinement pictures of gallery $\mathcal{G}_{\text{circlesMixed}}$. Some of the pictures in $\mathcal{G}_{\text{circlesFirst}}$ are shown in Figure 4.4. The gallery $\mathcal{G}_{\text{circlesFirst}}$ is generated by BCPG $G_{\text{circlesFirst:bag}} = (V_{\text{N}}, V_{\text{T}}, P', (S,\sigma), [8]\,, (1,0,0,0,0,0,0,0,0))$ where $V_{\text{N}} = \{S,T,U,F,C\}$, $V_{\text{T}} = \{b,w,g\}$ and P' is the set of rules in Figure 4.5.

Grammar $G_{\text{circlesFirst:bag}}$ is a modified version of grammar $G_{\text{circlesMixed:bag}}$. $G_{\text{circlesFirst:bag}}$ is made by changing only the value of the last position of the upper limit of Rule 4.1 from 71 to -1. This change ensures BCPG $G_{\text{circlesFirst:bag}}$ subdivides the initial square exactly once.

$\mathcal{G}_{\text{circlesFirst}}$ can also be generated by RCPG $G_{\text{circlesFirst}} = (V_{\text{N}}, V_{\text{T}}, P, (S,\sigma))$

$$
\begin{aligned}
S &\to [T,T,T,T,w,T,T,T,T] \\
T &\to U\,(\emptyset;\{S,T''\}) \mid \\
&\quad\ T''\,(\emptyset;\{U\}) \\
U &\to S'\,(\emptyset;\{T\}) \\
S' &\to [T',T',T',T',w,T',T',T',T']\,(\emptyset;\{U\}) \\
T' &\to U'\,(\emptyset;\{S',T''\}) \mid \\
&\quad\ T''\,(\emptyset;\{S',U\}) \\
U' &\to S''\,(\emptyset;\{T'\}) \\
S'' &\to [T'',T'',T'',T'',w,T'',T'',T'',T'']\,(\emptyset;\{U\}) \\
T'' &\to F\,(\emptyset;\{S,S',S'',U,U',F\}) \mid \\
&\quad\ C\,(\{F\};\{B,B',B'',B''',B'''',D,D',D'',T,T',S',S''\}) \\
F &\to C\,(\emptyset;\{T''\}) \\
C &\to B\,(\emptyset;\{B,T'',D''\}) \mid \\
&\quad\ B'\,(\{B\};\{B'\}) \mid \\
&\quad\ B''\,(\{B'\};\{B''\}) \mid \\
&\quad\ B'''\,(\{B''\};\{B'''\}) \mid \\
&\quad\ B''''\,(\{B'''\};\{B''''\}) \mid \\
&\quad\ D\,(\{B''''\};\{D\}) \mid \\
&\quad\ D'\,(\{D\};\{D'\}) \mid \\
&\quad\ D''\,(\{D'\};\{D''\}) \\
B &\to b\,(\{D''\};\{T''\}) \\
B' &\to b\,(\emptyset;\{B\}) \\
B'' &\to b\,(\emptyset;\{B,B'\}) \\
B''' &\to b\,(\emptyset;\{B,B',B''\}) \\
B'''' &\to b\,(\emptyset;\{B,B',B'',B'''\}) \\
D &\to g\,(\emptyset;\{B,B',B'',B''',B''''\}) \\
D' &\to g\,(\emptyset;\{B,B',B'',B''',B'''',D\}) \\
D'' &\to g\,(\emptyset;\{B,B',B'',B''',B'''',D,D'\})
\end{aligned}
$$

Figure 4.3: Rules for grammar $G_{\text{circlesMixed}}$

where $V_{\mathrm{N}} = \{S, T, F, C, B, B', B'', B''', B'''', D, D', D''\}$, $V_{\mathrm{T}} = \{w,\ b, g\}$ and P is the set of rules in Figure 4.6.

Gallery $\mathcal{G}_{\mathrm{circlesFirst}}$ was generated using RCPG $G_{\mathrm{circlesFirst}}$ after making the following changes to RCPG $G_{\mathrm{circlesMixed}}$:

- removed six variables ($S', S'', T', T'', U,\ U'$), and
- removed eight rules that serve no purpose.

It was easier to modify $G_{\mathrm{circlesMixed:bag}}$ to $G_{\mathrm{circlesFirst:bag}}$ as we only changed one value than it was to modify $G_{\mathrm{circlesMixed}}$ to $G_{\mathrm{circlesFirst}}$ which required many changes. Although $G_{\mathrm{circlesFirst}}$ has fewer variables and rules than $G_{\mathrm{circlesMixed}}$, it still has more variables and rules than its corresponding BCPG $G_{\mathrm{circlesFirst:bag}}$.

The following example generates only pictures that are in the second refinement.

Example 4.3. Consider gallery $\mathcal{G}_{\mathrm{circlesSecond}}$ which consists of structurally similar pictures to Figure 4.1b. This gallery has only the second refinement pictures of gallery $\mathcal{G}_{\mathrm{circlesMixed}}$. Some pictures in $\mathcal{G}_{\mathrm{circlesSecond}}$ are shown in Figure 4.7.

The gallery $\mathcal{G}_{\mathrm{circlesSecond}}$ is generated by BCPG $G_{\mathrm{circlesSecond:bag}} = (V_{\mathrm{N}}, V_{\mathrm{T}}, P', (S, \sigma), [8], (1, 0, 0, 0, 0, 0, 0, 0, 0))$ where $V_{\mathrm{N}} = \{S, T, U, F, C\}$, $V_{\mathrm{T}} = \{b, w, g\}$ and P' is the set of rules in Figure 4.8.

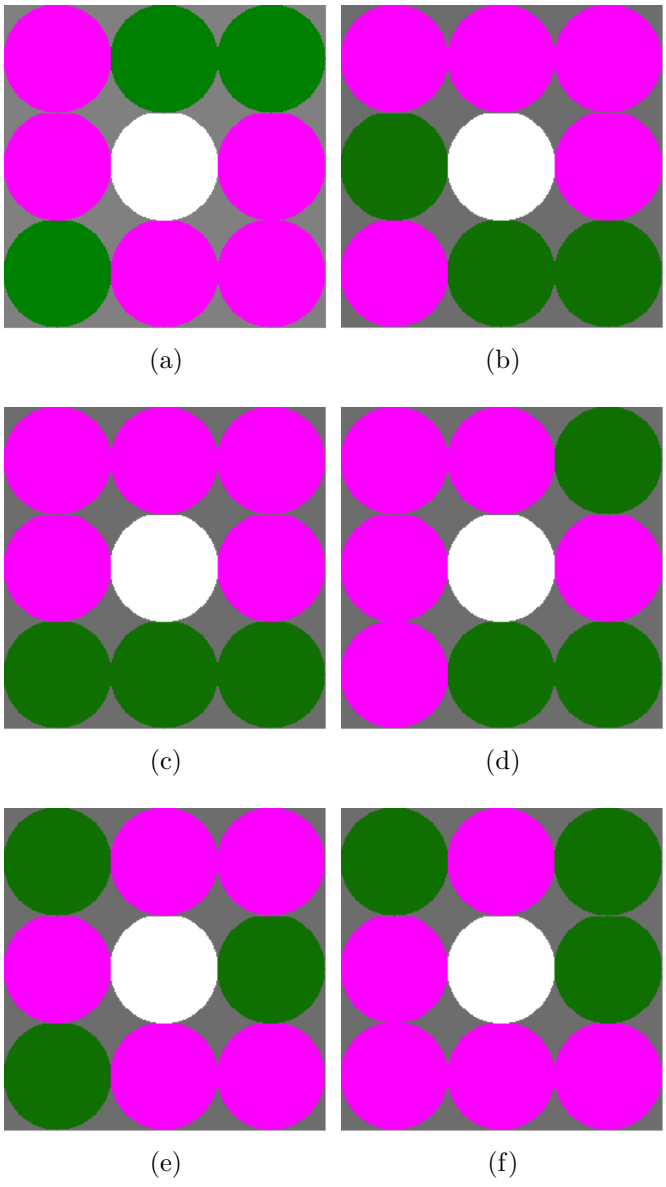
Grammar $G_{\mathrm{circlesSecond:bag}}$ is a modified version of grammar $G_{\mathrm{circlesMixed:bag}}$ which generates only the second refinement pictures of grammar $G_{\mathrm{circlesMixed:bag}}$. This was achieved by changing the following in $G_{\mathrm{circlesMixed:bag}}$:

- modifying the value of the last position of the upper limit of the Rule 4.1 from 71 to 7 (see Rule 4.5) , allowing subdivision of the initial square only twice, and
- modifying the value of the last position of the lower limit of Rule 4.3 from 0 to 8 (see Rule 4.6), to allow colouring once the picture has been subdivided twice.

These modifications of the two rules mean that only the second smallest pictures of gallery $\mathcal{G}_{\mathrm{circlesMixed}}$ are generated.

$\mathcal{G}_{\mathrm{circlesSecond}}$ can also be generated by RCPG $G_{\mathrm{circlesSecond}} = (V_{\mathrm{N}}, V_{\mathrm{T}}, P, (S, \sigma))$ where $V_{\mathrm{N}} = \{S, S', T, T', U, F, C, B, B', B'', B''',\ B'''', D, D', D''\}$, $V_{\mathrm{T}} = \{w, b, g\}$ and P is the set of rules in Figure 4.9.

For RCPG $G_{\mathrm{circlesSecond}}$ to generate only the second refinement pictures, we have modified the following in grammar $G_{\mathrm{circlesMixed}}$:



(a) (b)

(c) (d)

(e) (f)

Figure 4.4: Some pictures in $\mathcal{G}_{\text{circlesFirst}}$

$$
\begin{aligned}
S \rightarrow\ & [T,T,T,T,w,T,T,T,T]\,((1,0,0,0,0,0,0,0,0),\\
& (\infty,\infty,0,\infty,\infty,\infty,\infty,\infty);(-1,8,0,0,0,0,0,0,0))\\
T \rightarrow\ & U\,((0,1,0,0,0,0,0,0,0)\,,(0,\infty,\infty,0,\infty,\infty,\infty,-1)\,;\\
& \qquad (0,-1,1,0,0,0,0,1))\ \mid\\
& F\,((0,1,0,0,0,0,0,0,0)\,,(0,\infty,0,0,\infty,\infty,\infty,0)\,;\\
& \qquad (0,-1,0,1,0,0,0,0))\ \mid\\
& C\,((0,1,0,1,0,0,0,0,0)\,,\infty;(0,-1,0,0,1,0,0,0))\\
U \rightarrow\ & S\,((0,0,1,0,0,0,0,0,0)\,,(\infty,0,\infty,\infty,\infty,\infty,\infty,\infty)\,;\\
& \qquad (1,0,-1,0,0,0,0,0))\\
F \rightarrow\ & C\,((0,0,0,1,0,0,0,0,0)\,,(\infty,0,\infty,\infty,\infty,\infty,\infty,\infty)\,;\\
& \qquad (0,0,0,-1,1,0,0,0))\\
C \rightarrow\ & b\,((0,0,0,0,1,0,0,0)\,,(\infty,\infty,\infty,\infty,\infty,4,\infty,\infty)\,;\\
& \qquad (0,0,0,0,-1,1,0,0))\ \mid\\
& g\,((0,0,0,0,1,5,0,0)\,,(\infty,\infty,\infty,\infty,\infty,\infty,2,\infty)\,;\\
& \qquad (0,0,0,0,-1,0,1,0))\ \mid\\
& C\,((0,0,0,0,1,5,3,0)\,,\infty;(0,0,0,0,0,-5,-3,0))
\end{aligned}
\tag{4.4}
$$

Figure 4.5: Rules for grammar $G_{\text{circlesFirst:bag}}$

$$
\begin{aligned}
S &\to [T,T,T,T,w,T,T,T,T] \\
T &\to F\,(\emptyset;\{S,F\}) \mid \\
&\quad\; C\,(\{F\};\{\{B,B',B'',B''',B'''',D,D',D''\}\}) \mid \\
F &\to C\,(\emptyset;\{T\}) \\
C &\to B\,(\emptyset;\{B,T,D''\}) \mid \\
&\quad\; B'\,(\{B\};\{B'\}) \mid \\
&\quad\; B''\,(\{B'\};\{B''\}) \mid \\
&\quad\; B'''\,(\{B''\};\{B'''\}) \mid \\
&\quad\; B''''\,(\{B'''\};\{B''''\}) \mid \\
&\quad\; D\,(\{B''''\};\{D\}) \mid \\
&\quad\; D'\,(\{D\};\{D'\}) \mid \\
&\quad\; D''\,(\{D'\};\{D''\}) \\
B &\to b\,(\{D''\};\{T\}) \\
B' &\to b\,(\emptyset;\{B\}) \\
B'' &\to b\,(\emptyset;\{B,B'\}) \\
B''' &\to b\,(\emptyset;\{B,B',B''\}) \\
B'''' &\to b\,(\emptyset;\{B,B',B'',B'''\}) \\
D &\to g\,(\emptyset;\{B,B',B'',B''',B''''\}) \\
D' &\to g\,(\emptyset;\{B,B',B'',B''',B'''',D\}) \\
D'' &\to g\,(\emptyset;\{B,B',B'',B''',B'''',D,D'\})
\end{aligned}
$$

Figure 4.6: Rules for grammar $G_{\text{circlesFirst}}$

- removed three variable, S'', T'' and U', and
- removed five rules that are no longer needed.

In this example, we have observed that it was easier to modify BCPG than its corresponding RCPG because it required fewer changes and it did not involve addition or removal of rules and/or variables.

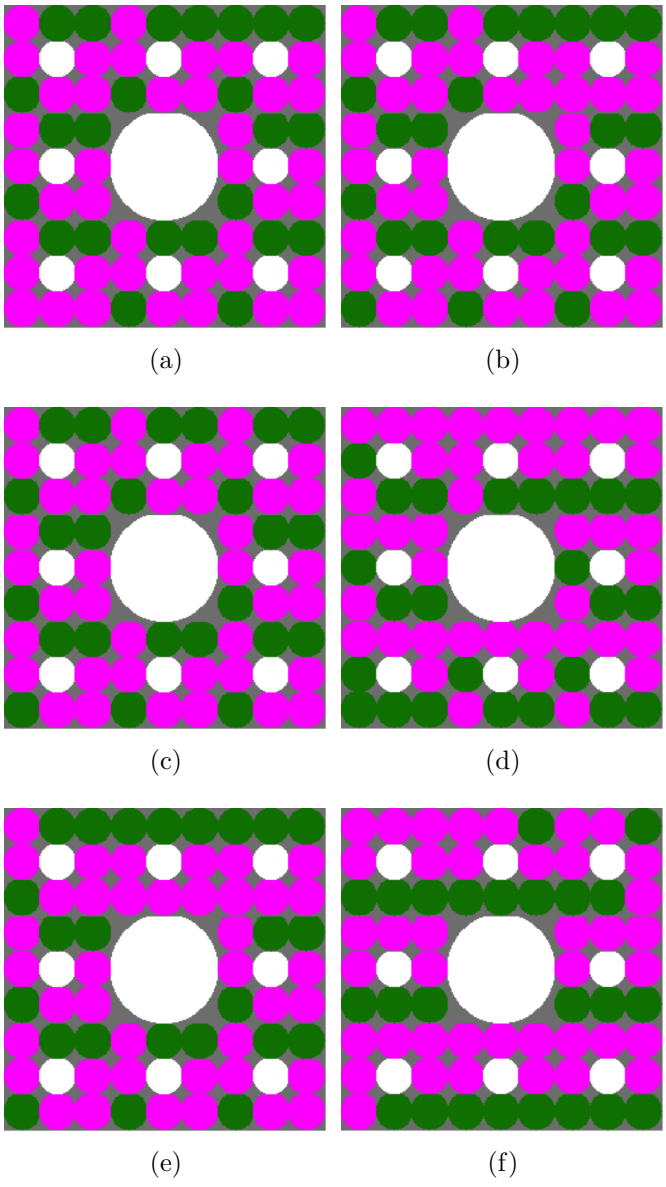


Figure 4.7: Some pictures in $\mathcal{G}_{\text{circlesSecond}}$

We now give an example that generates only pictures that are in the third refinement.

$$
\begin{aligned}
S \rightarrow\ & [T,T,T,T,w,T,T,T,T]\,((1,0,0,0,0,0,0,0,0),\\
& (\infty,\infty,0,\infty,\infty,\infty,\infty,\infty);(-1,8,0,0,0,0,0,0,0))\\
T \rightarrow\ & U\,((0,1,0,0,0,0,0,0,0)\,,(0,\infty,\infty,0,\infty,\infty,\infty,7)\,;\\
& \qquad (0,-1,1,0,0,0,0,1))\ \mid \qquad (4.5)\\
& F\,((0,1,0,0,0,0,0,0,8)\,,(0,\infty,0,0,\infty,\infty,\infty,0)\,;\\
& \qquad (0,-1,0,1,0,0,0,0))\ \mid \qquad (4.6)\\
& C\,((0,1,0,1,0,0,0,0,0)\,,\infty;(0,-1,0,0,1,0,0,0))\\
U \rightarrow\ & S\,((0,0,1,0,0,0,0,0,0)\,,(\infty,0,\infty,\infty,\infty,\infty,\infty,\infty)\,;\\
& \qquad (1,0,-1,0,0,0,0,0))\\
F \rightarrow\ & C\,((0,0,0,1,0,0,0,0,0)\,,(\infty,0,\infty,\infty,\infty,\infty,\infty,\infty)\,;\\
& \qquad (0,0,0,-1,1,0,0,0))\\
C \rightarrow\ & b\,((0,0,0,0,1,0,0,0)\,,(\infty,\infty,\infty,\infty,\infty,4,\infty,\infty)\,;\\
& \qquad (0,0,0,0,-1,1,0,0))\ \mid\\
& g\,((0,0,0,0,1,5,0,0)\,,(\infty,\infty,\infty,\infty,\infty,\infty,2,\infty)\,;\\
& \qquad (0,0,0,0,-1,0,1,0))\ \mid\\
& C\,((0,0,0,0,1,5,3,0)\,,\infty;(0,0,0,0,0,-5,-3,0))
\end{aligned}
$$

Figure 4.8: Rules for grammar $G_{\text{circlesSecond:bag}}$

$$
\begin{aligned}
S &\rightarrow [T,T,T,T,w,T,T,T,T] \\
T &\rightarrow U\,(\emptyset;\{S,T'\}) \\
U &\rightarrow S'\,(\emptyset;\{T\}) \\
S' &\rightarrow [T',T',T',T',w,T',T',T',T']\,(\emptyset;\{U\}) \\
T' &\rightarrow F\,(\emptyset;\{S,S',U,F\}) \mid \\
&\quad\; C\,(\{F\};\{B,B',B'',B''',B'''',D,D',D'',T,S'\}) \\
F &\rightarrow C\,(\emptyset;\{T'\}) \\
C &\rightarrow B\,(\emptyset;\{B,T',D''\}) \mid \\
&\quad\; B'\,(\{B\};\{B'\}) \mid \\
&\quad\; B''\,(\{B'\};\{B''\}) \mid \\
&\quad\; B'''\,(\{B''\};\{B'''\}) \mid \\
&\quad\; B''''\,(\{B'''\};\{B''''\}) \mid \\
&\quad\; D\,(\{B''''\};\{D\}) \mid \\
&\quad\; D'\,(\{D\};\{D'\}) \mid \\
&\quad\; D''\,(\{D'\};\{D''\}) \\
B &\rightarrow b\,(\{D''\};\{T'\}) \\
B' &\rightarrow b\,(\emptyset;\{B\}) \\
B'' &\rightarrow b\,(\emptyset;\{B,B'\}) \\
B''' &\rightarrow b\,(\emptyset;\{B,B',B''\}) \\
B'''' &\rightarrow b\,(\emptyset;\{B,B',B'',B'''\}) \\
D &\rightarrow g\,(\emptyset;\{B,B',B'',B''',B''''\}) \\
D' &\rightarrow g\,(\emptyset;\{B,B',B'',B''',B'''',D\}) \\
D'' &\rightarrow g\,(\emptyset;\{B,B',B'',B''',B'''',D,D'\})
\end{aligned}
$$

Figure 4.9: Rules for grammar $G_{\text{circlesSecond}}$

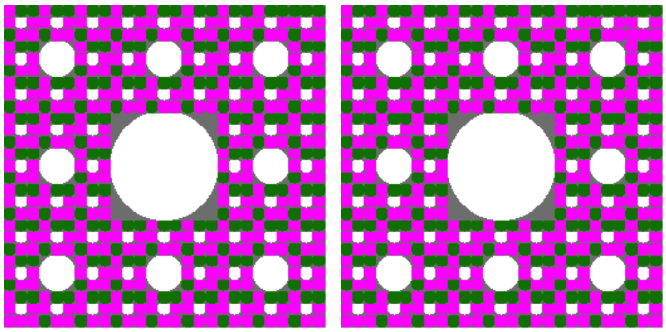
Example 4.4. Consider gallery $\mathcal{G}_{\text{circlesThird}}$ which consists of structurally similar pictures to Figure 4.1f. This gallery contains only the third refinement pictures in gallery $\mathcal{G}_{\text{circlesMixed}}$. Some pictures in $\mathcal{G}_{\text{circlesThird}}$ are shown in Figure 4.10. The gallery $\mathcal{G}_{\text{circlesThird}}$ is generated by BCPG $G_{\text{circlesThird:bag}} = (V_{\text{N}}, V_{\text{T}}, P', (S, \sigma), [8], (1, 0, 0, 0, 0, 0, 0, 0))$ where $V_{\text{N}} = \{S, T, U, F, C\}$, $V_{\text{T}} = \{b, w, g\}$ and P' is the set of rules in Figure 4.11.

The BCPG $G_{\text{circlesThird:bag}}$ is a modified version of grammar $G_{\text{circlesMixed:bag}}$ which generates the third refinement pictures of grammar $G_{\text{circlesMixed:bag}}$. This was achieved by changing the value of the last position of the lower limit in $G_{\text{circlesMixed:bag}}$ from 0 to 72 in Rule 4.7. This ensures that the colouring of pictures only starts after the initial square has been subdivided thrice.

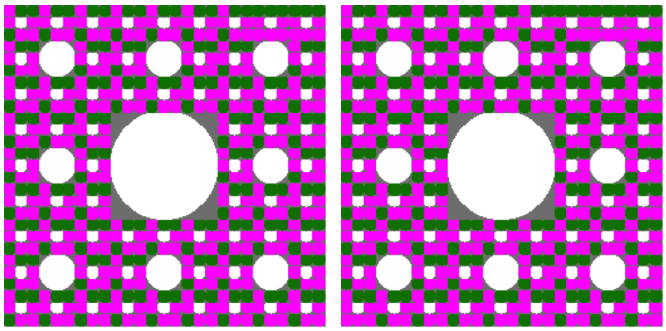
The gallery $\mathcal{G}_{\text{circlesThird}}$ can also be generated by RCPG $G_{\text{circlesThird}} = (V_{\text{N}}, V_{\text{T}}, P, (S, \sigma))$ where $V_{\text{N}} = \{S, S', S'', T, T', T'', U, U', F, C, B, B', B'', B''', B'''', D, D', D''\}$, $V_{\text{T}} = \{w, b, g\}$ and P is the set of rules in Figure 4.12.

RCPG $G_{\text{circlesThird}}$ is a modified version of RCPG $G_{\text{circlesMixed}}$. In this, we have removed two rules in RCPG $G_{\text{circlesMixed}}$.

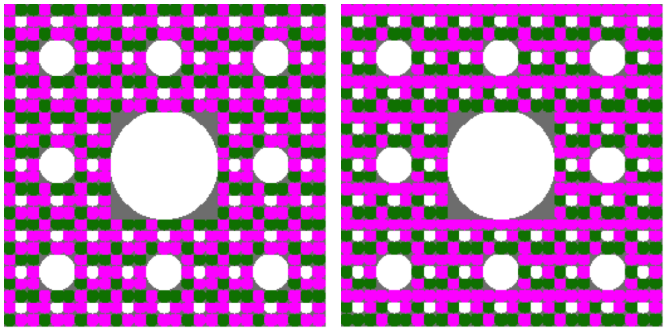
For this example, we observed that BCPG $G_{\text{circlesThird:bag}}$ has fewer rules and variables than RCPG $G_{\text{circlesThird}}$. RCPG $G_{\text{circlesThird}}$ needed more rules and variables to generate the gallery $\mathcal{G}_{\text{circlesThird}}$. This shows it is easier to write and modify BCPGs than RCPGs.



(a) (b)



(c) (d)



(e) (f)

Figure 4.10: Some pictures in $\mathcal{G}_{\text{circlesThird}}$

$$
\begin{aligned}
S \rightarrow\ & [T,T,T,T,w,T,T,T,T]\,((1,0,0,0,0,0,0,0,0),\\
& (\infty,\infty,0,\infty,\infty,\infty,\infty,\infty);(-1,8,0,0,0,0,0,0,0))\\
T \rightarrow\ & U\,((0,1,0,0,0,0,0,0,0)\,,(0,\infty,\infty,0,\infty,\infty,\infty,71)\,;\\
& \qquad (0,-1,1,0,0,0,0,1))\ \mid\\
& F\,((0,1,0,0,0,0,0,0,72)\,,(0,\infty,0,0,\infty,\infty,\infty,\infty)\,;\\
& \qquad (0,-1,0,1,0,0,0,0))\ \mid\\
& C\,((0,1,0,1,0,0,0,0,0)\,,\infty;(0,-1,0,0,1,0,0,0))\\
U \rightarrow\ & S\,((0,0,1,0,0,0,0,0,0)\,,(\infty,0,\infty,\infty,\infty,\infty,\infty,\infty)\,;\\
& \qquad (1,0,-1,0,0,0,0,0))\\
F \rightarrow\ & C\,((0,0,0,1,0,0,0,0,0)\,,(\infty,0,\infty,\infty,\infty,\infty,\infty,\infty)\,;\\
& \qquad (0,0,0,-1,1,0,0,0))\\
C \rightarrow\ & b\,((0,0,0,0,1,0,0,0)\,,(\infty,\infty,\infty,\infty,\infty,4,\infty,\infty)\,;\\
& \qquad (0,0,0,0,-1,1,0,0))\ \mid\\
& g\,((0,0,0,0,1,5,0,0)\,,(\infty,\infty,\infty,\infty,\infty,\infty,2,\infty)\,;\\
& \qquad (0,0,0,0,-1,0,1,0))\ \mid\\
& C\,((0,0,0,0,1,5,3,0)\,,\infty;(0,0,0,0,0,-5,-3,0))
\end{aligned}
\tag{4.7}
$$

Figure 4.11: Rules for grammar $G_{\text{circlesThird:bag}}$

$$
\begin{aligned}
S &\rightarrow [T,T,T,T,w,T,T,T,T] \\
T &\rightarrow U\,(\emptyset;\{S,T''\}) \\
U &\rightarrow S'\,(\emptyset;\{T\}) \\
S' &\rightarrow [T',T',T',T',w,T',T',T',T']\,(\emptyset;\{U\}) \\
T' &\rightarrow U'\,(\emptyset;\{S',T''\}) \\
U' &\rightarrow S''\,(\emptyset;\{T'\}) \\
S'' &\rightarrow [T'',T'',T'',T'',w,T'',T'',T'',T'']\,(\emptyset;\{U\}) \\
T'' &\rightarrow F\,(\emptyset;\{S,S',S'',U,U',F\}) \mid \\
&\quad\; C\,(\{F\};\{B,B',B'',B''',B'''',D,D',D'',T,T',S',S''\}) \\
F &\rightarrow C\,(\emptyset;\{T"\}) \\
C &\rightarrow B\,(\emptyset;\{B,T'',D''\}) \mid \\
&\quad\; B'\,(\{B\};\{B'\}) \mid \\
&\quad\; B''\,(\{B'\};\{B''\}) \mid \\
&\quad\; B'''\,(\{B''\};\{B'''\}) \mid \\
&\quad\; B''''\,(\{B'''\};\{B''''\}) \mid \\
&\quad\; D\,(\{B''''\};\{D\}) \mid \\
&\quad\; D'\,(\{D\};\{D'\}) \mid \\
&\quad\; D''\,(\{D'\};\{D''\}) \\
B &\rightarrow b\,(\{D''\};\{T''\}) \\
B' &\rightarrow b\,(\emptyset;\{B\}) \\
B'' &\rightarrow b\,(\emptyset;\{B,B'\}) \\
B''' &\rightarrow b\,(\emptyset;\{B,B',B''\}) \\
B'''' &\rightarrow b\,(\emptyset;\{B,B',B'',B'''\}) \\
D &\rightarrow g\,(\emptyset;\{B,B',B'',B''',B''''\}) \\
D' &\rightarrow g\,(\emptyset;\{B,B',B'',B''',B'''',D\}) \\
D'' &\rightarrow g\,(\emptyset;\{B,B',B'',B''',B'''',D,D'\})
\end{aligned}
$$

Figure 4.12: Rules for grammar $G_{\text{circlesThird}}$

4.2.2 Limiting Refinement of Sub-Pictures

In this section, we generate structurally similar pictures to a selected picture, with the main feature being the number of refinements of some sub-pictures. We also examine the ease of writing and amending RCPGs and BCPGs in this situation. We provide three examples: Example 4.5 generates pictures with the following structure: each quarter can be subdivided from zero to three times independently. Example 4.6 generates pictures with the top left quarter and the bottom right quarter subdivided exactly thrice, while the remaining quarters may be subdivided from zero to two times. Example 4.7 generates pictures with the top left quarter and the bottom right quarter subdivided exactly twice, while the remaining quarters may be subdivided from zero to two times. The following is the discussion of our findings:

Example 4.5. Consider the gallery $\mathcal{G}_{\text{flowerMixed}}$, which generates black and white flower-like pictures. Some pictures of the gallery $\mathcal{G}_{\text{flowerMixed}}$ are shown in Figure 4.13. Figure 4.13a is the smallest picture in the gallery $\mathcal{G}_{\text{flowerMixed}}$ structured as follows: the picture is subdivided into four equal-sized sub-squares. Each sub-square has four white circles in the centre on a grey background, surrounded by purple circles on a grey background. Let us call this picture π.

Every other picture in the gallery $\mathcal{G}_{\text{flowerMixed}}$ is structured as follows; the picture is subdivided into four equal-sized sub-squares. Any quarter may be divided again into four sub-squares. For the resulting picture, this may be repeated up to two times for any quarter that has its edge at the centre of the picture as a whole then π is placed in each quarter of each sub-picture.

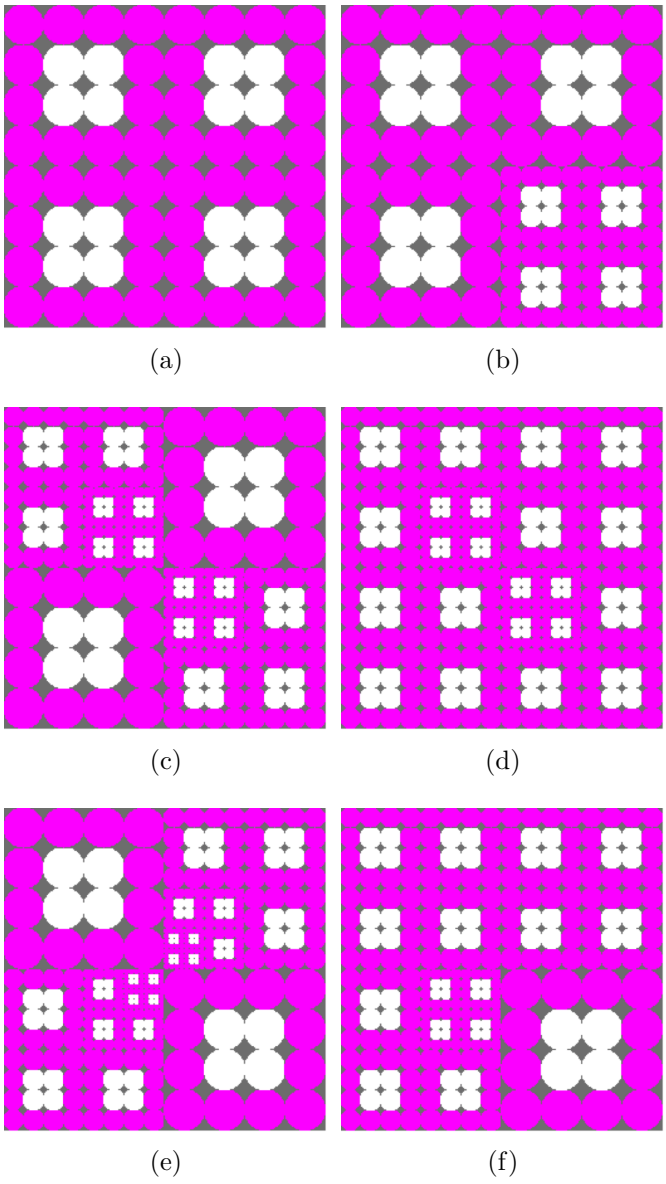
$\mathcal{G}_{\text{flowerMixed}}$ is generated by BCPG $G_{\text{flowerMixed:bag}} = (V_{\text{N}}, V_{\text{T}}, P', (S, \sigma), [9], 0)$ where $V_{\text{N}} = \{S, A, B, D, E, C, F\}$, $V_{\text{T}} = \{b, w\}$ and P' is the set of rules in Figure 4.14. The terminals b and w represents purple and white circles.

The bag positions in $G_{\text{flowerMixed:bag}}$ are used as follows:

- positions 1 and 2 count the occurrences of variables A and B respectively,
- positions 3 and 4 count the number of times Rules 4.8 and 4.13 have been applied,
- positions 5, 6 and 7 count the occurrences of variables F, D and E respectively, and
- positions 8 and 9 count the number of times Rules 4.9 and 4.11 respectively have been applied.

$\mathcal{G}_{\text{flowerMixed}}$ is also generated by the RCPG $G_{\text{flowerMixed}} = (V_{\text{N}}, V_{\text{T}}, P, (S, \sigma))$ where $V_{\text{N}} = \{S, A, A', A'', A''', B, B', B'', B''', D, D', D'', D''', E, E', E'', E''', C, F\}$, $V_{\text{T}} = \{w, b\}$ and P is the set of rules in Figure 4.15.

RCPG $G_{\text{flowerMixed}}$ generates the gallery $\mathcal{G}_{\text{flowerMixed}}$ but it uses more variables and rules. The following additional variables are in $G_{\text{flowerMixed}}$: $A', A'', A''', B', B'', B''', D'D'', D''', E', E'', E'''$. The variables were added to limit the number of times each square may be subdivided. There are twenty more rules in $G_{\text{flowerMixed}}$ than in $G_{\text{flowerMixed:bag}}$.



(a) (b) (c) (d) (e) (f)

Figure 4.13: Some pictures in $\mathcal{G}_{\text{flowerMixed}}$

Example 4.6. In this example, we consider the gallery $\mathcal{G}_{\text{flowerThree}}$ in Figure 4.16 which consists of similar pictures to Figure 4.13e (see Figure 4.13). Figure 4.16a is the smallest picture in the gallery. The structure of this picture is as follows: The top left and bottom right quarters are filled with

$$
\begin{aligned}
S \rightarrow\ & [A, D, E, B]\,(0, \infty; (1, 1, 0, 0, 0, 1, 1, 0, 0)) \\
A \rightarrow\ & [C, C, C, A]\,((1, 0, 0, 0, 0, 1, 1, 0, 0)\,, (\infty, \infty, 2, \infty, \infty, \infty, \infty, \infty, \infty)\,; \\
& \qquad (0, 0, 1, 0, 0, 0, 0, 0, 0))\ \mid && (4.8) \\
& F\,((1, 0, 0, 0, 0, 0, 0, 0, 0)\,, \infty; (-1, 0, 0, 0, 1, 0, 0, 0, 0)) \\
D \rightarrow\ & [C, C, D, C]\,((0, 0, 0, 0, 0, 0, 1, 0, 0)\,, (\infty, \infty, \infty, \infty, \infty, \infty, \infty, 2, \infty)\,; \\
& \qquad (0, 0, 0, 0, 0, 0, 0, 0, 1, 0))\ \mid && (4.9) \\
& F\,((0, 0, 0, 0, 0, 1, 0, 0, 0)\,, \infty; (0, 0, 0, 0, 1, -1, 0, 0, 0)) && (4.10) \\
E \rightarrow\ & [C, E, C, C]\,((0, 0, 0, 0, 0, 0, 0, 1, 0, 0)\,, (\infty, \infty, \infty, \infty, \infty, \infty, \infty, \infty, 2)\,; \\
& \qquad (0, 0, 0, 0, 0, 0, 0, 0, 0, 1))\ \mid && (4.11) \\
& F\,((0, 0, 0, 0, 0, 0, 0, 1, 0, 0)\,, \infty; (0, 0, 0, 0, 1, 0, -1, 0, 0)) && (4.12) \\
B \rightarrow\ & [B, C, C, C]\,((0, 1, 1, 0, 0, 0, 0, 0, 0, 0)\,, (\infty, \infty, \infty, 2, \infty, \infty, \infty, \infty, \infty)\,; \\
& \qquad (0, 0, 0, 1, 0, 0, 0, 0, 0, 0))\ \mid && (4.13) \\
& F\,((0, 1, 0, 0, 0, 0, 0, 0, 0, 0)\,, \infty; (0, -1, 0, 0, 1, 0, 0, 0, 0)) \\
F \rightarrow\ & C\,((0, 0, 0, 0, 1, 0, 0, 0, 0)\,, \infty; (0, 0, 0, 0, -1, 0, 0, 0, 0)) \\
C \rightarrow\ & [b, b, b, b, b, w, w, b, b, w, w, b, b, b, b, b]
\end{aligned}
$$

Figure 4.14: Rules for grammar $G_{\text{flowerMixed:bag}}$

pi. While the top right quarter and the bottom left quarter are subdivided thrice and π is placed in each quarter of each sub-picture.

Every other picture in gallery $\mathcal{G}_{\text{flowerThree}}$ is structured as follows: The top right quarter and the bottom left quarter can be subdivided from zero to two times and π is placed in each sub-picture. The top left and bottom right quarters of the picture cannot be subdivided any further. The gallery $\mathcal{G}_{\text{flowerThree}}$ is generated by BCPG $G_{\text{flowerThree:bag}} = (V_{\text{N}}, V_{\text{T}}, P', (S, \sigma), [9]\,, 0)$ where $V_{\text{N}} = \{S, A, B, D, E, C, F\}$, $V_{\text{T}} = \{b, w\}$ and P' is the set of rules in Figure 4.17.

$G_{\text{flowerThree:bag}}$ is a modified version of RCPG $G_{\text{flowerMixed:bag}}$. $\mathcal{G}_{\text{flowerThree}}$ was generated using BCPG $G_{\text{flowerThree:bag}}$ after making the following changes to $G_{\text{flowerMixed:bag}}$:

- changed the value of the third position of the upper limit of Rule 4.8 from 2 to 1,
- changed the value of the eighth position of the lower limit of Rule 4.10 from 0 to 3,
- changed the value of the last position of the lower limit of Rule 4.12 from 0 to 3,

$$
\begin{aligned}
S &\rightarrow [A, D, E, B] \\
A &\rightarrow [C, C, C, A'] \mid \\
&\quad F && (4.14)\\
A' &\rightarrow [C, C, C, A''] \mid \\
&\quad F && (4.15)\\
A'' &\rightarrow [C, C, C, A'''] \mid && (4.16)\\
&\quad F \\
A''' &\rightarrow F && (4.17)\\
D &\rightarrow [C, C, D', C] \mid \\
&\quad F && (4.18)\\
D' &\rightarrow [C, C, D'', C] \mid \\
&\quad F && (4.19)\\
D'' &\rightarrow [C, C, D''', C] \mid && (4.20)\\
&\quad F && (4.21)\\
D''' &\rightarrow F && (4.22)\\
E &\rightarrow [C, E', C, C] \mid \\
&\quad F && (4.23)\\
E' &\rightarrow [C, E'', C, C] \mid \\
&\quad F && (4.24)\\
E'' &\rightarrow [C, E''', C, C] \mid && (4.25)\\
&\quad F && (4.26)\\
E''' &\rightarrow F && (4.27)\\
B &\rightarrow [B', C, C, C] \mid \\
&\quad F && (4.28)\\
B' &\rightarrow [B'', C, C, C] \mid \\
&\quad F && (4.29)\\
B'' &\rightarrow [B''',' C, C, C] \mid && (4.30)\\
&\quad F && (4.31)\\
B''' &\rightarrow F && (4.32)\\
F &\rightarrow C \\
C &\rightarrow [b, b, b, b, b, w, w, b, b, w, w, b, b, b, b, b]
\end{aligned}
$$

Figure 4.15: Rules for grammar $G_{\text{flowerMixed}}$

- changed the value of the fourth position of the upper limit of Rule 4.13 from 2 to 1.

Both $G_{\text{flowerThree:bag}}$ and $G_{\text{flowerMixed:bag}}$ have the same number of variables and rules; we only change four values of the bag.

The gallery $\mathcal{G}_{\text{flowerThree}}$ can also be generated by RCPG $G_{\text{flowerThree}} = (V_{\text{N}}, V_{\text{T}}, P, (S, \sigma))$ where $V_{\text{N}} = \{S, A, A', A'', B, B', B'', D, D', D'', D''', E, E', E'', E''', C, F\}$, $V_{\text{T}} = \{w, b\}$ and P is the set of rules in Figure 4.18.

RCPG $G_{\text{flowerThree}}$ is a modified version of $G_{\text{flowerMixed}}$. RCPG $G_{\text{flowerThree}}$ was created by making the following modification to $G_{\text{flowerMixed}}$:

- removed two variables, A''' and B''',
- removed ten rules, namely Rules 4.16–4.19, 4.21, 4.23 4.24, 4.26, 4.30 and 4.32.

It can be observed that to generate gallery $\mathcal{G}_{\text{flowerThree}}$, BCPG $G_{\text{flowerThree:bag}}$ requires fewer rules and variables than for RCPG $G_{\text{flowerThree}}$. Moreover, fewer changes were required for BCPG $G_{\text{flowerThree:bag}}$ than RCPG $G_{\text{flowerThree}}$. Writing and modifying the BCPG was easier than the RCPG in this example.

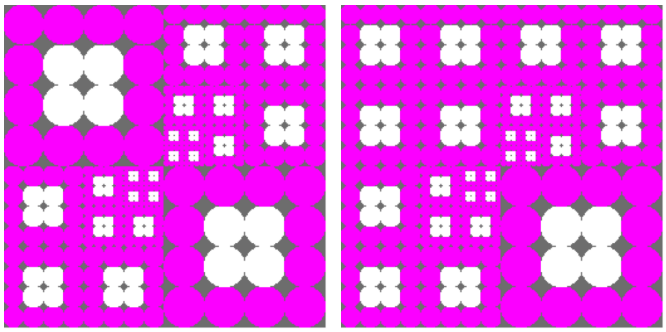
Example 4.7. Consider the gallery $\mathcal{G}_{\text{flowerTwo}}$ in Figure 4.19, which consists of structurally similar pictures to Figure 4.13c (see Figure 4.13). Figure 4.19a is the smallest picture in the gallery. It has the following structure: The top left and the bottom right quarter of the picture are filled with π. While the top right quarter and the bottom left quarter are subdivided twice and π is placed in each sub-picture.

Every other picture in the gallery $\mathcal{G}_{\text{flowerTwo}}$ is structured as follows; the top right and bottom left quarters of the picture cannot be subdivided any further. The top left and the bottom right quarters can be subdivided from zero to two times and π is placed in each sub-picture.

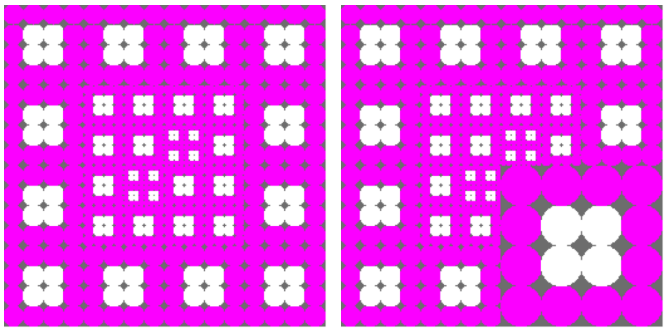
The gallery $\mathcal{G}_{\text{flowerTwo}}$ is generated by BCPG $G_{\text{flowerTwo:bag}} = (V_{\text{N}}, V_{\text{T}}, P', (S, \sigma), [9], 0)$ where $V_{\text{N}} = \{S, A, B, D, E, C, F\}$, $V_{\text{T}} = \{b, w\}$ and P' is the set of rules in Figure 4.20.

$G_{\text{flowerTwo:bag}}$ is a modified version of BCPG $G_{\text{flowerMixed:bag}}$. $\mathcal{G}_{\text{flowerTwo}}$ was generated by BCPG $G_{\text{flowerTwo:bag}}$ after making the following changes to $G_{\text{flowerMixed:bag}}$:

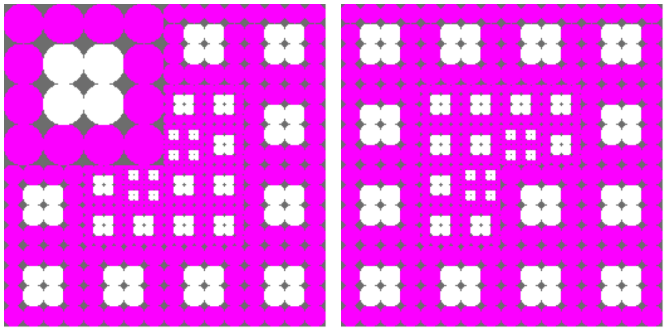
- changed the value of the third position of the upper limit of Rule 4.8 from 2 to 1,
- changed the value of the eighth position of the upper limit of Rule 4.9 from 2 to 1,



(a) (b)



(c) (d)



(e) (f)

Figure 4.16: Some pictures in $\mathcal{G}_{\text{flowerThree}}$

$$
\begin{aligned}
S &\rightarrow [A, D, E, B]\,(0, \infty; (1, 1, 0, 0, 0, 1, 1, 0, 0)) \\
A &\rightarrow [C, C, C, A]\,((1, 0, 0, 0, 0, 1, 1, 0, 0)\,, (\infty, \infty, 1, \infty, \infty, \infty, \infty, \infty, \infty)\,; \\
&\qquad\quad (0, 0, 1, 0, 0, 0, 0, 0, 0))\ \mid \\
&\qquad F\,((1, 0, 0, 0, 0, 0, 0, 0, 0)\,, \infty; (-1, 0, 0, 0, 1, 0, 0, 0, 0)) \\
D &\rightarrow [C, C, D, C]\,((0, 0, 0, 0, 0, 0, 1, 0, 0)\,, (\infty, \infty, \infty, \infty, \infty, \infty, \infty, 2, \infty)\,; \\
&\qquad\quad (0, 0, 0, 0, 0, 0, 0, 0, 1, 0))\ \mid \\
&\qquad F\,((0, 0, 0, 0, 0, 1, 0, 3, 0)\,, \infty; (0, 0, 0, 0, 1, -1, 0, 0, 0)) \\
E &\rightarrow [C, E, C, C]\,((0, 0, 0, 0, 0, 0, 0, 1, 0, 0)\,, (\infty, \infty, \infty, \infty, \infty, \infty, \infty, \infty, 2)\,; \\
&\qquad\quad (0, 0, 0, 0, 0, 0, 0, 0, 0, 1))\ \mid \\
&\qquad F\,((0, 0, 0, 0, 0, 0, 1, 0, 3)\,, \infty; (0, 0, 0, 0, 1, 0, -1, 0, 0)) \\
B &\rightarrow [B, C, C, C]\,((0, 1, 1, 0, 0, 0, 0, 0, 0)\,, (\infty, \infty, \infty, 1, \infty, \infty, \infty, \infty, \infty)\,; \\
&\qquad\quad (0, 0, 0, 1, 0, 0, 0, 0, 0))\ \mid \\
&\qquad F\,((0, 1, 0, 0, 0, 0, 0, 0, 0)\,, \infty; (0, -1, 0, 0, 1, 0, 0, 0, 0)) \\
F &\rightarrow C\,((0, 0, 0, 0, 1, 0, 0, 0, 0)\,, \infty; (0, 0, 0, 0, -1, 0, 0, 0, 0)) \\
C &\rightarrow [b, b, b, b, b, w, w, b, b, w, w, b, b, b, b, b]
\end{aligned}
$$

Figure 4.17: Rules for grammar $G_{\text{flowerThree:bag}}$

- changed the value of the last position of the upper limit of Rule 4.11 from 2 to 1,
- changed the value of the fourth position of the upper limit of Rule 4.13 from 2 to 1,
- changed the value of the eighth position of the lower limit of Rule 4.10 from 0 to 2,
- changed the value of the last position of the lower limit of Rule 4.12 from 0 to 2.

Both BCPGs $G_{\text{flowerTwo:bag}}$ and $G_{\text{flowerMixed:bag}}$ have the same number of variables and rules, the only difference is some changed values in the upper and lower limits of the rules.

The gallery $\mathcal{G}_{\text{flowerTwo}}$ can also be generated by RCPG $G_{\text{flowerTwo}} = (V_{\text{N}}, V_{\text{T}}, P, (S, \sigma))$ where $V_{\text{N}} = \{S, A, A', A'', B, B', B'', D, D', D'', E, E', E'', C, F\}$, $V_{\text{T}} = \{w, b\}$ and P is the set of rules in Figure 4.21.

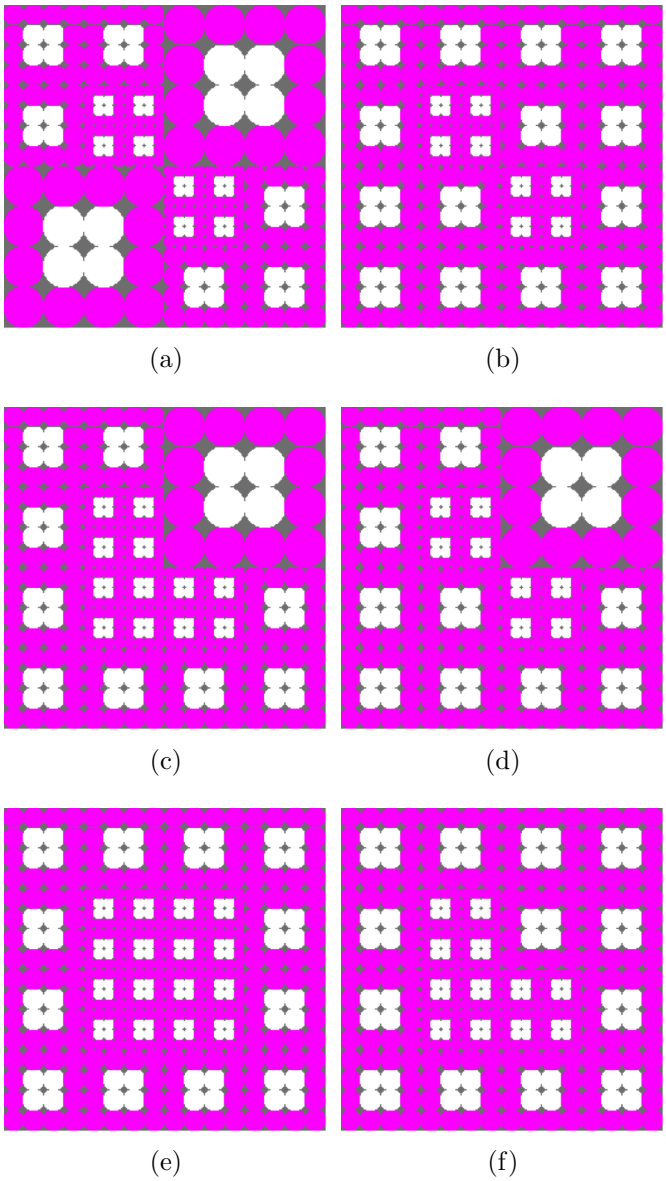
$G_{\text{flowerTwo}}$ is a modified version of grammar $G_{\text{flowerMixed}}$. $\mathcal{G}_{\text{flowerTwo}}$ was generated by BCPG $G_{\text{flowerTwo}}$ after making the following changes to $G_{\text{flowerMixed}}$:

$$
\begin{aligned}
S &\rightarrow [A, D, E, B] \\
A &\rightarrow [C, C, C, A']\,(\emptyset; \{E, E', D, D'\}) \mid \\
&\quad\ F \\
A' &\rightarrow [C, C, C, A''] \mid \\
&\quad\ F \\
A'' &\rightarrow F \\
D &\rightarrow [C, C, D', C] \\
D' &\rightarrow [C, C, D'', C] \\
D'' &\rightarrow [C, C, D''', C] \mid \\
D''' &\rightarrow F \\
E &\rightarrow [C, E', C, C] \\
E' &\rightarrow [C, E'', C, C] \\
E'' &\rightarrow [C, E''', C, C] \mid \\
E''' &\rightarrow F \\
B &\rightarrow [B', C, C, C]\,(\emptyset; \{E, E', D, D'\}) \mid \\
&\quad\ F \\
B' &\rightarrow [B'', C, C, C] \mid \\
&\quad\ F \\
B'' &\rightarrow F \\
F &\rightarrow C \\
C &\rightarrow [b, b, b, b, b, w, w, b, b, w, w, b, b, b, b, b]
\end{aligned}
$$

Figure 4.18: Rules for grammar $G_{\text{flowerThree}}$

- four variables, A''', D''', E''' and B''' were removed,
- twelve rules, namely Rules 4.16–4.20, 4.22–4.25, 4.27, 4.30 and 4.32 were removed.

It was easier to modify BCPG $G_{\text{flowerMixed:bag}}$ and write BCPG $G_{\text{flowerTwo:bag}}$ than to modify RCPG $G_{\text{flowerMixed}}$ and write RCPG $G_{\text{flowerTwo}}$ as fewer changes were needed in the BCPG than the RCPG. Moreover, to generate the gallery $\mathcal{G}_{\text{flowerTwo}}$, BCPG $G_{\text{flowerTwo:bag}}$ required fewer variables and rules than its corresponding RCPG $G_{\text{flowerTwo}}$.



(a) (b) (c) (d) (e) (f)

Figure 4.19: Some pictures in $\mathcal{G}_{\text{flowerTwo}}$

$$
\begin{aligned}
S &\rightarrow [A, D, E, B]\,(0, \infty; (1, 1, 0, 0, 0, 1, 1, 0, 0)) \\
A &\rightarrow [C, C, C, A]\,((1, 0, 0, 0, 0, 1, 1, 0, 0)\,, (\infty, \infty, 1, \infty, \infty, \infty, \infty, \infty, \infty)\,; \\
&\qquad\quad (0, 0, 1, 0, 0, 0, 0, 0, 0))\ \mid \\
&\quad\ F\,((1, 0, 0, 0, 0, 0, 0, 0, 0)\,, \infty; (-1, 0, 0, 0, 1, 0, 0, 0, 0)) \\
D &\rightarrow [C, C, D, C]\,((0, 0, 0, 0, 0, 1, 0, 0, 0)\,, (\infty, \infty, \infty, \infty, \infty, \infty, \infty, 1, \infty)\,; \\
&\qquad\quad (0, 0, 0, 0, 0, 0, 0, 1, 0))\ \mid \\
&\quad\ F\,((0, 0, 0, 0, 0, 1, 0, 2, 0)\,, \infty; (0, 0, 0, 0, 1, -1, 0, 0, 0)) \\
E &\rightarrow [C, E, C, C]\,((0, 0, 0, 0, 0, 0, 1, 0, 0)\,, (\infty, \infty, \infty, \infty, \infty, \infty, \infty, \infty, 1)\,; \\
&\qquad\quad (0, 0, 0, 0, 0, 0, 0, 0, 1))\ \mid \\
&\quad\ F\,((0, 0, 0, 0, 0, 0, 1, 0, 2)\,, \infty; (0, 0, 0, 0, 1, 0, -1, 0, 0)) \\
B &\rightarrow [B, C, C, C]\,((0, 1, 1, 0, 0, 0, 0, 0, 0)\,, (\infty, \infty, \infty, 1, \infty, \infty, \infty, \infty, \infty)\,; \\
&\qquad\quad (0, 0, 0, 1, 0, 0, 0, 0, 0))\ \mid \\
&\quad\ F\,((0, 1, 0, 0, 0, 0, 0, 0, 0)\,, \infty; (0, -1, 0, 0, 1, 0, 0, 0, 0)) \\
F &\rightarrow C\,((0, 0, 0, 0, 1, 0, 0, 0, 0)\,, \infty; (0, 0, 0, 0, -1, 0, 0, 0, 0)) \\
C &\rightarrow [b, b, b, b, b, w, w, b, b, w, w, b, b, b, b, b]
\end{aligned}
$$

Figure 4.20: Rules for grammar $G_{\text{flowerTwo:bag}}$

$$
\begin{aligned}
S &\rightarrow [A, D, E, B] \\
A &\rightarrow [C, C, C, A']\,(\emptyset; \{E, E', D, D'\}) \mid \\
&\quad\ F \\
A' &\rightarrow [C, C, C, A''] \mid \\
&\quad\ F \\
A'' &\rightarrow F \\
D &\rightarrow [C, C, D', C] \\
D' &\rightarrow [C, C, D'', C] \\
D'' &\rightarrow F \\
E &\rightarrow [C, E', C, C] \\
E' &\rightarrow [C, E'', C, C] \\
E'' &\rightarrow F \\
B &\rightarrow [B', C, C, C]\,(\emptyset; \{E, E', D, D'\}) \mid \\
&\quad\ F \\
B' &\rightarrow [B'', C, C, C] \mid \\
&\quad\ F \\
B'' &\rightarrow F \\
F &\rightarrow C \\
C &\rightarrow [b, b, b, b, b, w, w, b, b, w, w, b, b, b, b, b]
\end{aligned}
$$

Figure 4.21: Rules for grammar $G_{\text{flowerTwo}}$

4.2.3 Limiting Both Refinement Levels and Refinement of Sub-Pictures

This section provides examples of grammars that do both the limiting of the number of refinements of the entire picture and limiting refinements of sub-pictures. The combining of these two strategies was done to learn if the similarity of the generated pictures will improve. We provide four examples, where the first example provides a gallery that consists of pictures in the first, second and third levels of refinement. The second example provides a BCPG and RCPG which generate pictures in the second level of refinement. Moreover, the third and fourth examples both provide pictures that are in the third level of refinement.

Example 4.8. Consider gallery $\mathcal{G}_{\text{carpetMixed}}$, which consists of a variation on the sequence of pictures approximating the Sierpiński carpet [1]. Some

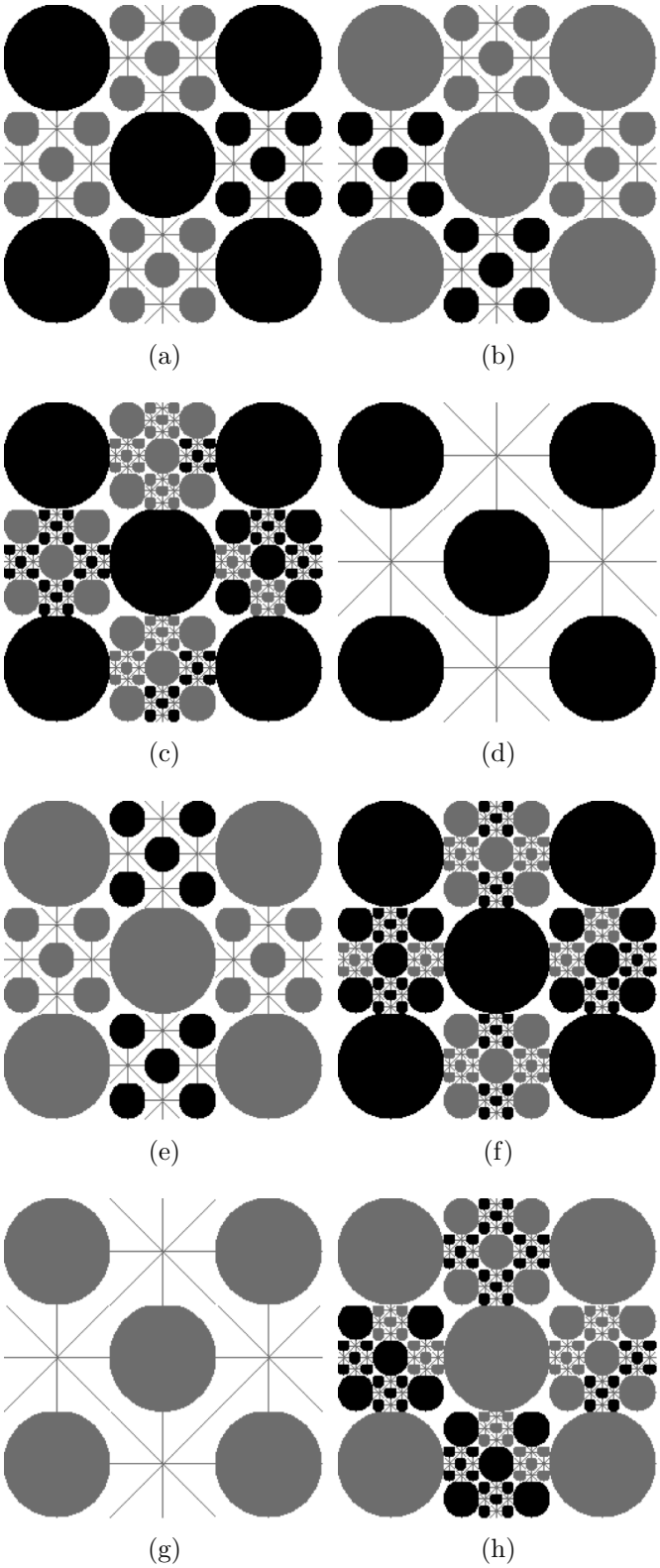


Figure 4.22: Some pictures in $\mathcal{G}_{\text{carpetMixed}}$

pictures in $\mathcal{G}_{\text{carpetMixed}}$ are shown in Figure 4.22. The gallery consists of pictures from the first up to the third refinement level only. This was done to have sub-pictures that are easily visible.

Figures 4.22d and 4.22g are the first level of refinement pictures of gallery $\mathcal{G}_{\text{carpetMixed}}$ and have the following format: A picture is divided into nine equal-sized sub-squares. The sub-squares in the corners and the centre are all filled with either grey or black circles on a white background. The remaining sub-squares are filled with a grey star on a white background.

The second level of refinement pictures in $\mathcal{G}_{\text{carpetMixed}}$ (examples shown in Figures 4.22a and 4.22b) have the following format: Every picture has the circles as in either in Figure 4.22d or Figure 4.22g. The remaining squares are filled with a copy of either Figure 4.22d or Figure 4.22g.

Furthermore, the third level of refinement pictures in $\mathcal{G}_{\text{carpetMixed}}$ (examples shown in Figures 4.22f and 4.22h) have the following format: The black and grey circles are as in Figure 4.22d and 4.22g. Each of the remaining four squares are subdivided like pictures in Figures 4.22a and 4.22b but the order of grey and black circles might vary.

The gallery $\mathcal{G}_{\text{carpetMixed}}$ is generated by BCPG $G_{\text{carpetMixed:bag}} = (V_{\text{N}}, V_{\text{T}}, P', (S, \sigma), [7], (1, 0, 0, 0, 0, 0, 0, 0))$ where $V_{\text{N}} = \{S, T, U, F\}$, $V_{\text{T}} = \{b, w, g\}$ and P' is the set of rules in Figure 4.23. Terminals w and g represent black and grey circles, respectively and b represents a grey star.

The bag positions 1 to 4 in $G_{\text{carpetMixed:bag}}$ count the occurrences of variables S, T, U and F respectively. Positions 5 and 6 keep a counter of the application of Rules 4.33–4.36, ensuring that there is alternation in colouring between black and grey circles. The last position counts the number of times Rule 4.37 has been applied so as to put a lower and higher limit on the application of the rules.

The gallery $\mathcal{G}_{\text{carpetMixed}}$ is also generated by RCPG $G_{\text{carpetMixed}} = (V_{\text{N}}, V_{\text{T}}, P, (S, \sigma))$ where $V_{\text{N}} = \{S, S', \ldots, S'^{(23)}, T, T', \ldots, T'''', U, U', \ldots, U'''', F\}$, $S'^{(23)}$ stands for S with 23 primes, $V_{\text{T}} = \{b, w, g\}$ and P is the sketched set of rules in Figure 4.24. Terminals w and g represent black and grey circles, respectively and b represents a grey star.

For this example we have provided a sketched set of rules because the grammar uses many variables and rules and it can be very complicated to show all the rules and context. This grammar uses 33 variables and more than 33 rules. In an RCPG it is, therefore, possible to ensure that the initial square is not sub-divided more than a predetermined number of times, for example, three in this case, and alternating between rules. However, in general, the RCPG requires more variables and rules than the corresponding BCPG.

Next we present a gallery which consists of the pictures that are in the

$$
\begin{aligned}
S \rightarrow\ & [g,T,g,T,g,T,g,T,g]\,((1,0,0,0,0,0,0,0)\,,(\infty,\infty,0,\infty,0,\infty,\infty)\,;\\
& \quad (-1,4,0,0,1,0,0))\ \mid && (4.33)\\
& [g,T,g,T,g,T,g,T,g]\,((1,0,0,0,0,0,1,0)\,,(\infty,\infty,0,\infty,\infty,\infty,\infty)\,;\\
& \quad (-1,4,0,0,1,-1,0))\ \mid && (4.34)\\
& [w,T,w,T,w,T,w,T,w]\,((1,0,0,0,0,0,0,0)\,,(\infty,\infty,0,\infty,\infty,0,\infty)\,;\\
& \quad (-1,4,0,0,0,1,0))\ \mid && (4.35)\\
& [w,T,w,T,w,T,w,T,w]\,((1,0,0,0,1,0,0,0)\,,(\infty,\infty,0,\infty,\infty,\infty,\infty)\,;\\
& \quad (-1,4,0,0,-1,1,0)) && (4.36)\\
T \rightarrow\ & U\,((0,1,0,0,0,0,0,0)\,,(0,\infty,\infty,0,\infty,\infty,20)\,;\\
& \quad (0,-1,1,0,0,0,1))\ \mid && (4.37)\\
& F\,((0,1,0,0,0,0,0,0)\,,(0,\infty,0,0,\infty,\infty,\infty)\,;\\
& \quad (0,-1,0,1,0,0,0))\ \mid && (4.38)\\
& b\,((0,1,0,1,0,0,0)\,,\infty;(0,-1,0,0,0,0,0,0))\\
U \rightarrow\ & S\,((0,0,1,0,0,0,0,0)\,,(\infty,0,\infty,\infty,\infty,\infty,\infty)\,;(1,0,-1,0,0,0,0))\\
F \rightarrow\ & b\,((0,0,0,1,0,0,0)\,,(\infty,0,\infty,\infty,\infty,\infty,\infty)\,;(0,0,0,-1,0,0,0))
\end{aligned}
$$

Figure 4.23: Rules for grammar $G_{\text{carpetMixed:bag}}$

second level of refinement.

Example 4.9. Consider gallery $\mathcal{G}_{\text{carpetSecond}}$, which consists of only second refinement pictures of $\mathcal{G}_{\text{carpetMixed}}$. The gallery was modified to have similar pictures to Figure 4.22a. Some pictures in $\mathcal{G}_{\text{carpetSecond}}$ are shown in Figure 4.25.

The gallery $\mathcal{G}_{\text{carpetSecond}}$ is generated by BCPG $G_{\text{carpetSecond:bag}} = (V_{\text{N}}, V_{\text{T}}, P', (S,\sigma), [7]\,, (1,0,0,0,0,0,0,0))$ where $V_{\text{N}} = \{S,T,U,F\}$, $V_{\text{T}} = \{b,w,g\}$ and P' is the set of rules in Figure 4.26. Terminals w and g represent black and grey circles, respectively and b represents a grey star. $G_{\text{carpetSecond:bag}}$ is a modified version of grammar $G_{\text{carpetMixed:bag}}$ and generates only the second refinement pictures of $\mathcal{G}_{\text{carpetMixed}}$.

The bag positions in BCPG $G_{\text{carpetSecond:bag}}$ are used as in BCPG $G_{\text{carpetMixed:bag}}$. We have only changed the value of the last position of the upper limit of Rule 4.37 from 20 to 4 (see Rule 4.44), and the value of the last position of the lower limit of Rule 4.38 from 0 to 5 (see Rule 4.45).These changes ensure that only second refinement pictures are generated.

The gallery $\mathcal{G}_{\text{carpetSecond}}$ can also be generated by RCPG $G_{\text{carpetSecond}} = (V_{\text{N}}, V_{\text{T}}, P, (S,\sigma))$ where $V_{\text{N}} = \{S, S', S'', S''', S'''', S''''', T, T', T''U, U', F\}$, $V_{\text{T}} = \{b,w,g\}$ and P is the set of rules in Figure 4.27. Terminals w and g represent black and grey circles, respectively and b

$$
\begin{aligned}
S &\to [g,T,g,T,g,T,w,T,g]\,(\emptyset;\{U\}) \mid \\
&\quad [w,T',w,T',w,T',w,T',w]\,(\emptyset;\{U\}) && (4.39)\\
S' &\to [w,T'',w,T'',w,T'',w,T'',w]\,(\emptyset;\{U,U'\}) \\
S'' &\to [g,T''',g,T''',g,T''',w,T''',g]\,(\emptyset;\{U,U'\}) \\
S''' &\to [w,T'',w,T'',w,T'',w,T'',w]\,(\emptyset;\{U,U'\}) \\
S'''' &\to [g,T''',g,T''',g,T''',w,T''',g]\,(\emptyset;\{U,U'\}) \\
S''''' &\to [w,T'',w,T'',w,T'',w,T'',w]\,(\emptyset;\{U,U'\}) \\
S'''''' &\to [w,T'',w,T'',w,T'',w,T'',w]\,(\emptyset;\{U,U',\ldots\}) \\
&\quad \vdots \\
T &\to U\,(\emptyset;\{S,F\}) \mid \\
&\quad F\,(\emptyset;\{S,U,U',F\}) \mid && (4.40)\\
&\quad b\,(\emptyset;\{S\}) && (4.41)\\
T' &\to U'\,(\emptyset;\{S,F\}) \mid \\
&\quad F\,(\emptyset;\{S,U,U',F\}) \mid && (4.42)\\
&\quad b\,(\emptyset;\{S\}) && (4.43)\\
T'' &\to U''\,(\{T''\};\{S,S',\ldots,U,U',\ldots,F\}) \\
&\quad \vdots \\
U &\to S'\,(\emptyset;\{T,T',\ldots,S'\}) \mid \\
&\quad S''\,(\{S'\};\{T,T',\ldots,S''\}) \mid \\
&\quad S'''\,(\{S''\};\{T,T',\ldots,S'''\}) \mid \\
&\quad S''''\,(\{S'''\};\{T,T',\ldots,S''''\}) \\
U' &\to S''\,(\emptyset;\{T,T',\ldots,S''\}) \mid \\
&\quad S'''\,(\{S''\};\{T,T',\ldots,S'''\}) \mid \\
&\quad S''''\,(\{S'''\};\{T,T',\ldots,S''''\}) \mid \\
&\quad S'''''\,(\{S''''\};\{T,T',\ldots,S'''''\}) \\
U'' &\to S''''''\,(\emptyset;\{T,T',\ldots,S''\}) \mid \\
&\quad \vdots \\
F &\to b\,(\emptyset;\{T,T',T''\})
\end{aligned}
$$

Figure 4.24: Rules for grammar $G_{\text{carpetMixed}}$

represents a grey star. $G_{\text{carpetSecond}}$ is a modified version of grammar $G_{\text{carpetMixed}}$ and generates only the second refinement pictures of $\mathcal{G}_{\text{carpetMixed}}$.

For us to generate only the second level of refinement pictures, we removed some rules that allow colouring at first level of refinement (for example, Rules 4.40 and 4.41) so that colouring would start after the second refinement level.Moreover, we removed the rules that are of no use. We have also removed some variables that were not needed.

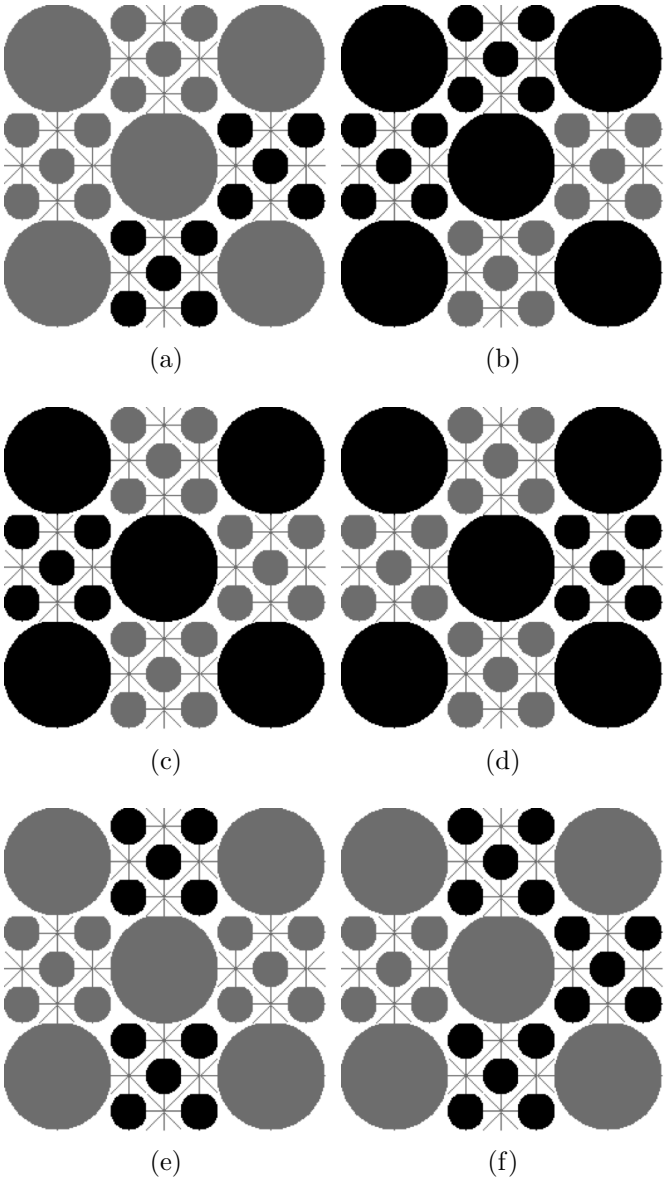


Figure 4.25: Some pictures in $\mathcal{G}_{\text{carpetSecond}}$

Example 4.10. Consider gallery $\mathcal{G}_{\text{carpetThird}}$, which consists of only the third refinement pictures of $\mathcal{G}_{\text{carpetMixed}}$. Some pictures of gallery $\mathcal{G}_{\text{carpetThird}}$ are

$$
\begin{aligned}
S \rightarrow\ & [g,T,g,T,g,T,g,T,g]\,((1,0,0,0,0,0,0,0)\,,(\infty,\infty,0,\infty,0,\infty,\infty)\,;\\
& \quad (-1,4,0,0,1,0,0))\ \mid\\
& [g,T,g,T,g,T,g,T,g]\,((1,0,0,0,0,0,1,0)\,,(\infty,\infty,0,\infty,\infty,\infty,\infty)\,;\\
& \quad (-1,4,0,0,1,-1,0))\ \mid\\
& [w,T,w,T,w,T,w,T,w]\,((1,0,0,0,0,0,0,0)\,,(\infty,\infty,0,\infty,\infty,0,\infty)\,;\\
& \quad (-1,4,0,0,0,1,0))\ \mid\\
& [w,T,w,T,w,T,w,T,w]\,((1,0,0,0,1,0,0,0)\,,(\infty,\infty,0,\infty,\infty,\infty,\infty)\,;\\
& \quad (-1,4,0,0,-1,1,0))\\
T \rightarrow\ & U\,((0,1,0,0,0,0,0,0)\,,(0,\infty,\infty,0,\infty,\infty,4)\,;\\
& \quad (0,-1,1,0,0,0,1))\ \mid \qquad (4.44)\\
& F\,((0,1,0,0,0,0,0,5)\,,(0,\infty,0,0,\infty,\infty,\infty)\,;\\
& \quad (0,-1,0,1,0,0,0))\ \mid \qquad (4.45)\\
& b\,((0,1,0,1,0,0,0)\,,\infty;(0,-1,0,0,0,0,0,0))\\
U \rightarrow\ & S\,((0,0,1,0,0,0,0,0)\,,(\infty,0,\infty,\infty,\infty,\infty,\infty)\,;(1,0,-1,0,0,0,0))\\
F \rightarrow\ & b\,((0,0,0,1,0,0,0)\,,(\infty,0,\infty,\infty,\infty,\infty,\infty)\,;(0,0,0,-1,0,0,0))
\end{aligned}
$$

Figure 4.26: Rules for grammar $G_{\text{carpetSecond:bag}}$

shown in Figure 4.28.

$\mathcal{G}_{\text{carpetThird}}$ is generated by BCPG $G_{\text{carpetThird:bag}} = (V_{\text{N}}, V_{\text{T}}, P', (S,\sigma), [7]\,, (1,0,0,0,0,0,0,0))$ where $V_{\text{N}} = \{S,T,U,F\}$, $V_{\text{T}} = \{b,w,g\}$ and P' is the set of rules in Figure 4.29. Terminals w and g represent black and grey circles, respectively and b represents a grey star. $G_{\text{carpetThird:bag}}$ is a modified version of BCPG $G_{\text{carpetMixed:bag}}$. $G_{\text{carpetThird:bag}}$ generates only the third level of refinement pictures of $G_{\text{carpetMixed:bag}}$.

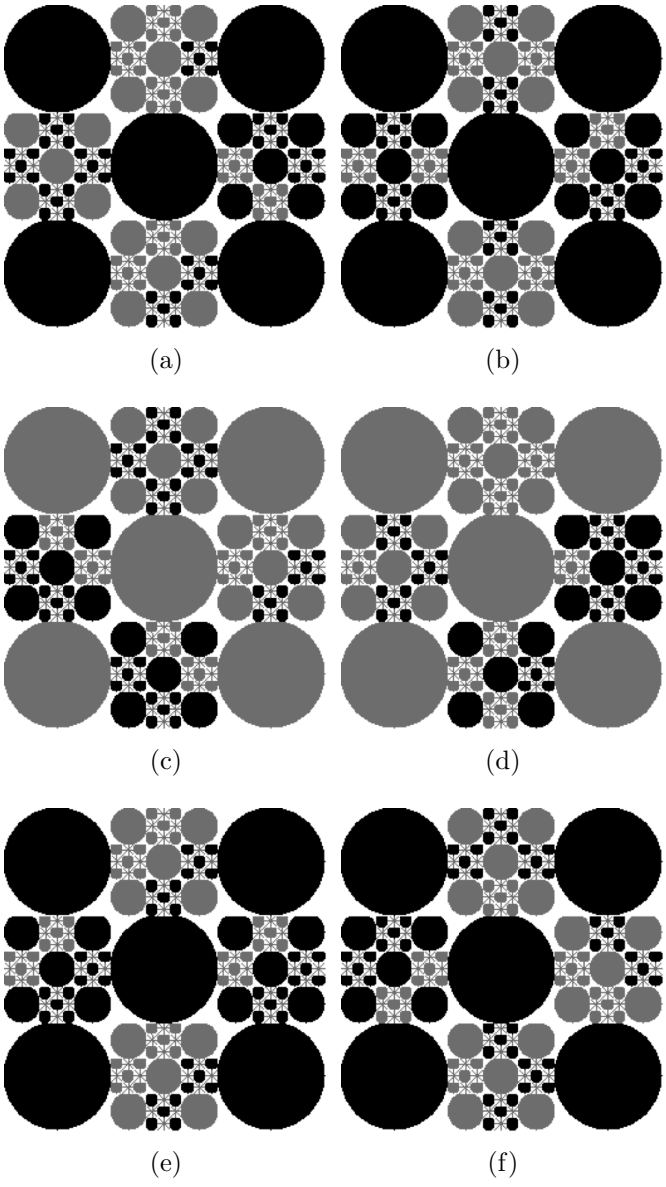
The gallery $\mathcal{G}_{\text{carpetThird}}$ can also be generated by RCPG $G_{\text{carpetThird}} = (V_{\text{N}}, V_{\text{T}}, P,\ (S,\sigma))$ where $V_{\text{N}} = \{S, S', \ldots, S'^{(23)}, T, T', \ldots, T'''', U, U', \ldots, U'''', F\}$, $V_{\text{T}} = \{b,w,g\}$ and P is the set of sketched rules in Figure 4.30. Terminals w and g represent black and grey circles, respectively and b represents a grey star.

The bag positions in BCPG $G_{\text{carpetThird:bag}}$ are as in BCPG $G_{\text{carpetMixed:bag}}$. For us to generate only the third level of refinement pictures, we only changed the last position of the lower limit of Rule 4.38 from 0 to 21 (see Rule 4.46). In the RCPG we could achieve this by removing some rules that allow colouring at lower refinement levels, for example, Rules 4.40, 4.41, 4.42 and 4.43 so that colouring can start after the third refinement.

We have provided a sketched set of rules for this example also because the grammar uses many variables and rules and it can be very complicated

$$
\begin{array}{lll}
S & \to & [g,T,g,T,g,T,w,T,g]\,(\emptyset;\{U\}) \mid \\
 & & [w,T',w,T',w,T',w,T',w]\,(\emptyset;\{U\}) \\
S' & \to & [w,T'',w,T'',w,T'',w,T'',w]\,(\emptyset;\{U,U'\}) \\
S'' & \to & [g,T'',g,T'',g,T'',w,T'',g]\,(\emptyset;\{U,U'\}) \\
S''' & \to & [w,T'',w,T'',w,T'',w,T'',w]\,(\emptyset;\{U,U'\}) \\
S'''' & \to & [g,T'',g,T'',g,T'',w,T'',g]\,(\emptyset;\{U,U'\}) \\
S''''' & \to & [w,T'',w,T'',w,T'',w,T'',w]\,(\emptyset;\{U,U'\}) \\
T & \to & U\,(\emptyset;\{S,F\}) \\
T' & \to & U'\,(\emptyset;\{S,F\}) \\
T'' & \to & F\,(\{T''\};\{S,U,U',F\}) \mid \\
 & & b\,(\emptyset;\{S\}) \\
U & \to & S'\,(\emptyset;\{T,T',T'',S'\}) \mid \\
 & & S''\,(\{S'\};\{T,T',T'',S''\}) \mid \\
 & & S'''\,(\{S''\};\{T,T',T'',S'''\}) \mid \\
 & & S''''\,(\{S'''\};\{T,T',T'',S''''\}) \\
U' & \to & S''\,(\emptyset;\{T,T',T'',S''\}) \mid \\
 & & S'''\,(\{S''\};\{T,T',T'',S'''\}) \mid \\
 & & S''''\,(\{S'''\};\{T,T',T'',S''''\}) \mid \\
 & & S'''''\,(\{S''''\};\{T,T',T'',S'''''\}) \\
F & \to & b\,(\emptyset;\{T,T',T''\})
\end{array}
$$

Figure 4.27: Rules for grammar $G_{\text{carpetSecond}}$



(a) (b)

(c) (d)

(e) (f)

Figure 4.28: Some pictures in $\mathcal{G}_{\text{carpetThird}}$

to show all the rules and context. This grammar uses 33 variables and more than 33 rules.

$$
\begin{aligned}
S \rightarrow\ & [g,T,g,T,g,T,g,T,g]\,((1,0,0,0,0,0,0,0)\,,(\infty,\infty,0,\infty,0,\infty,\infty)\,;\\
& \qquad (-1,4,0,0,1,0,0))\ \mid\\
& [g,T,g,T,g,T,g,T,g]\,((1,0,0,0,0,0,1,0)\,,(\infty,\infty,0,\infty,\infty,\infty,\infty)\,;\\
& \qquad (-1,4,0,0,1,-1,0))\ \mid\\
& [w,T,w,T,w,T,w,T,w]\,((1,0,0,0,0,0,0,0)\,,(\infty,\infty,0,\infty,\infty,0,\infty)\,;\\
& \qquad (-1,4,0,0,0,1,0))\ \mid\\
& [w,T,w,T,w,T,w,T,w]\,((1,0,0,0,1,0,0,0)\,,(\infty,\infty,0,\infty,\infty,\infty,\infty)\,;\\
& \qquad (-1,4,0,0,-1,1,0))\\
T \rightarrow\ & U\,((0,1,0,0,0,0,0,0)\,,(0,\infty,\infty,0,\infty,\infty,20)\,;\\
& \qquad (0,-1,1,0,0,0,1))\ \mid\\
& F\,((0,1,0,0,0,0,0,21)\,,(0,\infty,0,0,\infty,\infty,\infty)\,;\\
& \qquad (0,-1,0,1,0,0,0))\ \mid\\
& b\,((0,1,0,1,0,0,0)\,,\infty;(0,-1,0,0,0,0,0,0))\\
U \rightarrow\ & S\,((0,0,1,0,0,0,0,0)\,,(\infty,0,\infty,\infty,\infty,\infty,\infty)\,;(1,0,-1,0,0,0,0))\\
F \rightarrow\ & b\,((0,0,0,1,0,0,0)\,,(\infty,0,\infty,\infty,\infty,\infty,\infty)\,;(0,0,0,-1,0,0,0))
\end{aligned}
\tag{4.46}
$$

Figure 4.29: Rules for grammar $G_{\text{carpetThird:bag}}$

Example 4.11. Consider gallery $\mathcal{G}_{\text{carpetThirdgrey}}$, which consists of only the third refinement pictures of $\mathcal{G}_{\text{carpetMixed}}$ with all the circles in the first level of refinement grey. Some pictures of gallery $\mathcal{G}_{\text{carpetThirdgrey}}$ are shown in Figure 4.31. The pictures in this gallery have the same format of subdividing the picture and sub-pictures as gallery $\mathcal{G}_{\text{carpetThird}}$, except that in $\mathcal{G}_{\text{carpetThirdgrey}}$, the circles in the first level of refinement are only grey.

$\mathcal{G}_{\text{carpetThirdgrey}}$ is generated by BCPG $G_{\text{carpetThirdgrey:bag}} = (V_{\text{N}}, V_{\text{T}}, P', (S,\sigma), [7]\,, (1,0,0,0,0,0,0,0))$ where $V_{\text{N}} = \{S,T,U,F\}$, $V_{\text{T}} = \{b,w,g\}$ and P' is the set of rules in Figure 4.32. Terminals w and g represent black and grey circles, respectively and b represents a grey star. $G_{\text{carpetThirdgrey:bag}}$ is a modified version of BCPG $G_{\text{carpetMixed:bag}}$. $G_{\text{carpetThirdgrey:bag}}$ generates only the third level of refinement pictures of $G_{\text{carpetMixed:bag}}$ with all the circles in the first level of refinement grey.

$\mathcal{G}_{\text{carpetThirdgrey}}$ can also be generated by RCPG $G_{\text{carpetThirdgrey}} = (V_{\text{N}}, V_{\text{T}}, P, (S,\sigma))$ where $V_{\text{N}} = \{S, S', \ldots, S'^{(23)}, T, T', \ldots, T'''', U, U', \ldots, U'''', F\}$, $V_{\text{T}} = \{b,w,g\}$ and P is the set of sketched rules in Figure 4.33. Terminals

$$
\begin{array}{lll}
S & \to & [g,T,g,T,g,T,w,T,g]\,(\emptyset;\{U\}) \mid \\
 & & [w,T',w,T',w,T',w,T',w]\,(\emptyset;\{U\}) \\
S' & \to & [w,T'',w,T'',w,T'',w,T'',w]\,(\emptyset;\{U,U'\}) \\
S'' & \to & [g,T''',g,T''',g,T''',w,T''',g]\,(\emptyset;\{U,U'\}) \\
 & & \vdots \\
S''''' & \to & [w,T'',w,T'',w,T'',w,T'',w]\,(\emptyset;\{U,U',\ldots\}) \\
 & & \vdots \\
T & \to & U\,(\emptyset;\{S,F\}) \\
T' & \to & U'\,(\emptyset;\{S,F\}) \\
T'' & \to & U''\,(\{\}\,;\{S,S',\ldots,U,U',\ldots,F\}) \\
T''' & \to & U'''\,(\{\}\,;\{S,S',\ldots,U,U',\ldots,F\}) \\
T'''' & \to & F\,(\emptyset;\{S,S',S'',\ldots,U,U',\ldots,F\}) \mid \\
 & & b\,(\emptyset;\{S,S',\ldots\}) \\
U & \to & S'\,(\emptyset;\{T,T',\ldots,S'\}) \mid \\
 & & S''\,(\{S'\}\,;\{T,T',\ldots,S''\}) \mid \\
 & & S'''\,(\{S''\}\,;\{T,T',\ldots,S'''\}) \mid \\
 & & S''''\,(\{S'''\}\,;\{T,T',\ldots,S''''\}) \\
U' & \to & S''\,(\emptyset;\{T,T',\ldots,S''\}) \mid \\
 & & S'''\,(\{S''\}\,;\{T,T',\ldots,S'''\}) \mid \\
 & & S''''\,(\{S'''\}\,;\{T,T',\ldots,S''''\}) \mid \\
 & & S'''''\,(\{S''''\}\,;\{T,T',\ldots,S'''''\}) \\
U'' & \to & S'''''\,(\emptyset;\{T,T',\ldots,S''\}) \mid \\
 & & \vdots \\
F & \to & b\,(\emptyset;\{T,T',\ldots\})
\end{array}
$$

Figure 4.30: Rules for grammar $G_{\text{carpetThird}}$

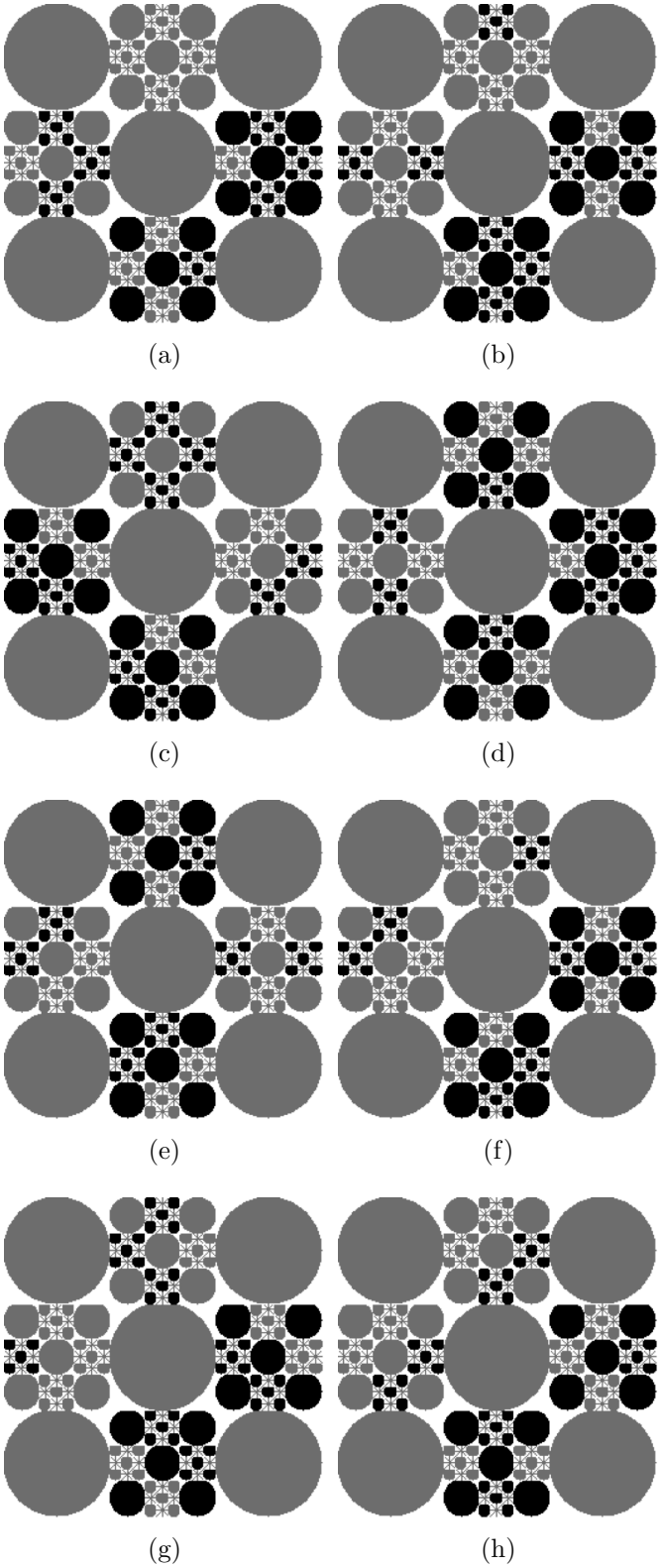


Figure 4.31: A selection of pictures in $\mathcal{G}_{\text{carpetThirdgrey}}$

$$
\begin{aligned}
S \rightarrow\ & [g,T,g,T,g,T,g,T,g]\,((1,0,0,0,0,0,0,0,0)\,,(\infty,\infty,0,\infty,0,\infty,\infty)\,;\\
& \quad (-1,4,0,0,1,0,0))\ \mid\\
& [g,T,g,T,g,T,g,T,g]\,((1,0,0,0,0,1,0)\,,(\infty,\infty,0,\infty,\infty,\infty,\infty)\,;\\
& \quad (-1,4,0,0,1,-1,0))\ \mid\\
& [w,T,w,T,w,T,w,T,w]\,((1,0,0,0,1,0,0)\,,(\infty,\infty,0,\infty,\infty,\infty,\infty)\,;\\
& \quad (-1,4,0,0,-1,1,0))\\
T \rightarrow\ & U\,((0,1,0,0,0,0,0)\,,(0,\infty,\infty,0,\infty,\infty,20)\,;\\
& \quad (0,-1,1,0,0,0,1))\ \mid\\
& F\,((0,1,0,0,0,0,0,21)\,,(0,\infty,0,0,\infty,\infty,\infty)\,;\\
& \quad (0,-1,0,1,0,0,0))\ \mid\\
& b\,((0,1,0,1,0,0,0)\,,\infty;(0,-1,0,0,0,0,0,0))\\
U \rightarrow\ & S\,((0,0,1,0,0,0,0)\,,(\infty,0,\infty,\infty,\infty,\infty,\infty)\,;(1,0,-1,0,0,0,0))\\
F \rightarrow\ & b\,((0,0,0,1,0,0,0)\,,(\infty,0,\infty,\infty,\infty,\infty,\infty)\,;(0,0,0,-1,0,0,0))
\end{aligned}
\tag{4.47}
$$

Figure 4.32: Rules for grammar $G_{\text{carpetThirdgrey:bag}}$

w and g represent black and grey circles, respectively and b represents a grey star.

The bag positions in BCPG $G_{\text{carpetThirdgrey:bag}}$ are as in BCPG $G_{\text{carpetMixed:bag}}$. For us to generate only the third level of refinement pictures with grey circles in the first refinement sub-pictures, we removed one rule and changed the last position of the lower limit of Rule 4.38 from 0 to 21 (see Rule 4.47). In the RCPG $G_{\text{carpetThirdgrey}}$ we could achieve this by removing Rule 4.39 and removing some rules that allows colouring at lower refinement levels, for example, Rules 4.40, 4.41, 4.42 and 4.43 to allow colouring only at after the third refinement level.

In an RCPG it is, therefore, possible to ensure that there is an alternation in the application of rules. However, in general, the RCPG requires more variables and rules than the corresponding BCPG. Moreover, every time we change the number of times a picture is refined the number of variables and/or rules also change, for example, from two to three, the number of variables and rules in the RCPG increases and the rules themselves may have to change, in particular, the random context. In the corresponding BCPG, only selected components of the lower and upper limits of some rules need to be replaced.

In the next section we review the approaches used in the generation of similar pictures.

$$
\begin{aligned}
S &\to [g,T,g,T,g,T,w,T,g]\,(\emptyset;\{U\})\\
S' &\to [w,T'',w,T'',w,T'',w,T'',w]\,(\emptyset;\{U,U'\})\\
S'' &\to [g,T''',g,T''',g,T''',w,T''',g]\,(\emptyset;\{U,U'\})\\
&\;\;\vdots\\
S''''' &\to [w,T'',w,T'',w,T'',w,T'',w]\,(\emptyset;\{U,U',\ldots\})\\
&\;\;\vdots\\
T &\to U\,(\emptyset;\{S,F\})\\
T' &\to U'\,(\emptyset;\{S,F\})\\
T'' &\to U''\,(\{\}\,;\{S,S',\ldots,U,U',\ldots,F\})\\
T''' &\to U'''\,(\{\}\,;\{S,S',\ldots,U,U',\ldots,F\})\\
T'''' &\to F\,(\emptyset;\{S,S',S'',\ldots,U,U',\ldots,F\})\;\mid\\
&\quad b\,(\emptyset;\{S,S',\ldots\})\\
U &\to S'\,(\emptyset;\{T,T',\ldots,S'\})\;\mid\\
&\quad S''\,(\{S'\}\,;\{T,T',\ldots,S''\})\;\mid\\
&\quad S'''\,(\{S''\}\,;\{T,T',\ldots,S'''\})\;\mid\\
&\quad S''''\,(\{S'''\}\,;\{T,T',\ldots,S''''\})\\
U' &\to S''\,(\emptyset;\{T,T',\ldots,S''\})\;\mid\\
&\quad S'''\,(\{S''\}\,;\{T,T',\ldots,S'''\})\;\mid\\
&\quad S''''\,(\{S'''\}\,;\{T,T',\ldots,S''''\})\;\mid\\
&\quad S'''''\,(\{S''''\}\,;\{T,T',\ldots,S'''''\})\\
U'' &\to S'''''\,(\emptyset;\{T,T',\ldots,S''\})\;\mid\\
&\;\;\vdots\\
F &\to b\,(\emptyset;\{T,T',\ldots\})
\end{aligned}
$$

Figure 4.33: Rules for grammar $G_{\text{carpetThirdgrey}}$

4.3 Review of the Approaches of Generation of Similar Pictures

This section provides a review of the approaches used for generation of structurally similar pictures.

4.3.1 Limiting Refinement Levels

In this method, we showed that it is possible to generate a finite number of structurally similar pictures. This is because only pictures belonging to a particular level are generated by the grammar. We also showed that the first level of refinement galleries contained fewer pictures than the following refinements due to the number of sub-pictures available and the possible number of variation in colouring. The possibility of the generation of more pictures increases with the increase in the number of times the initial square is sub-divided and or the number of colours used. Since it is possible to generate a finite number of pictures which are in a given level of refinement, creating structurally similar pictures for a visual password scheme is simplified as the need for creating many pictures and then finding the ones with similar features is minimized.

4.3.2 Limiting Refinement of Sub-Pictures

The pictures generated by limiting the refinement of sub-pictures have also been observed to produce a finite number of structurally similar images. The number of pictures produced depends on the allowable number of levels of refinement of individual sub-pictures. Fewer levels of refinement for sub-pictures result into fewer pictures produced, while allowing more levels of refinements results in an increase in the number of generated pictures. As pictures generated here have similar sub-pictures, the generated pictures tend to be structurally similar and thus obtaining similar pictures would not require generating many pictures.

4.3.3 Limiting Both Refinement Levels and Refinement of Sub-Pictures

Generating pictures that combine both limiting refinement levels and refinement of sub-pictures has also shown to produce a finite number of similar pictures. The galleries that contain first refinement levels consists of fewer pictures than galleries that consist of later levels of refinements. This is because there are fewer options of subdivision of a picture in the lower levels of refinement than in higher levels of refinements.

4.4 Conclusion

This chapter presented approaches to generating similar pictures that could be used as password pictures and distractors in a visual password system. We used bag context picture grammars and random context picture grammars to generate these pictures. Both BCPGs and RCPGs can be written and modified in many ways in order to generate structurally similar pictures. Three methods of producing structurally similar pictures were presented. These methods are limiting refinement levels of the generated pictures, limiting the refinement of some sub-pictures in the picture and limiting both the levels of refinements of the generated pictures and sub-pictures in the picture. Both BCPGs and RCPGs can be written and modified in many ways in order to generate structurally similar pictures. We have also compared RCPGs and BCPGs on the generation of structurally similar pictures by modifying the grammars. We have found both grammars to generate a finite number of pictures that are structurally similar.

In our examples, BCPGs require fewer rules and variables to generate similar pictures than RCPGs. It is easier to modify BCPGs than RCPGs as BCPGs need fewer changes to the grammar than RCPGs. Unlike BCPGs, RCPGs often require the addition or removal of rules and variables. Modifying RCPGs requires changing the rules and context while modifying BCPGs often only requires changing some bag values. Moreover, tracking changes in BCPGs is easier as there are fewer places to modify, usually the values of the bag, and often don't require the introduction of new rules or variables. The use of BCPGs saves time and requires fewer operations. Therefore, for our purposes, it seems that BCPGs are better than RCPGs when the grammar is written and modified to generate only structurally similar pictures.

Chapter 5

Estimating Similarity Mathematically

5.1 Introduction

Research in the generation of similar pictures is currently of interest as a result of applications that require the use of similar pictures such as visual password systems [62, 64]. To generate similar pictures efficiently, there is a need to determine effective methods of doing this in a controlled way. In the previous chapter, we presented several galleries of structurally similar pictures generated by BCPGs and RCPGs where different methods have been used to control the generation of pictures.

In this chapter, we measure the mathematical similarity of the pictures generated by BCPGs and RCPGs using several methods for improving the similarity of the pictures. Based on the results of this research, we recommend effective methods of generating similar pictures.

Many factors affect the similarity of pictures, such as the colours used, the size of sub-pictures in the pictures, the distribution of colours and of sub-pictures in the picture. In this research, we evaluated how the refinement of pictures and the distribution of sub-pictures and/or of the colours in the picture affect the similarity of syntactically generated pictures. Using the galleries of the pictures presented, we divided the galleries into three groups. In the first group, we calculated the mathematical similarity of the pictures to determine if pictures generated in the same level of refinement are more similar than pictures in different refinement levels.

We kept the refinement the same and only changed the distribution of colours. In the second group of pictures, we limited the number of refinements of certain sub-pictures of the picture and assessed if this improved the similarity of the generated pictures. Lastly we tested using both limiting levels of refinement and limiting refinement of sub-pictures.

A spatial colour distribution descriptor [6] was used in this research

to estimate the similarity of the generated pictures. Okundaye et al. [63] observed that this colour model provided excellent results in determining the similarity of syntactically generated pictures and, from our research, [45], we found that it correlated with the human perceptual view. We also used the Img(Rummager) application [5] to calculate the SpCD similarity of the pictures. The calculated SpCD values were also used to calculate the average similarity for each gallery which was used to compare the similarity of pictures between different galleries.

5.2 Mathematical Similarity Measure

One of the key elements in our research is to determine the mathematical similarity of the generated pictures. SpCD was used to measure the mathematical similarity of the generated pictures in this research. SpCD is a compact composite descriptor which combines colour and spatial colour distribution [6]. SpCD is suitable for coloured pictures that contain a small number of colours and texture regions like hand-drawn sketches and coloured graphics such as the ones generated by picture grammars.

SpCD uses a fuzzy system to extract colour information and maps the colours of the image into a custom 8-colours palette [6]. The system uses three channels of HSV (Hue Saturation Value) as input and outputs an 8-bin histogram, with each bin representing a preset colour. These preset colours are (0) Black, (1) Red, (2) Orange/Yellow, (3) Green, (4) Cyan, (5) Blue, (6) Magenta and (7) White. Eight tiles of size 4×4, with each tile corresponding to one of the preset colours, are created to include the spatial distribution information of the colour.

To estimate the similarity of the images, all the images in question are uploaded to the database. SpCD is applied to every image in the database, creating an XML file with the entries. To retrieve similarity results, the user enters a query image. The query image is exported as 8 tiles and creates a 128-dimensional vector. Then each image in the XML file is converted to a 128-dimensional vector and compared to the query image. The system then outputs a ranked list of images according to the similarity with the query image. Figure 5.1 portrays the process.

5.3 Determining the Similarity of Pictures

Determining the mathematical similarity of pictures is an essential factor in this research, not only because we want to ensure that the generated pictures are similar, but also because we want to determine the most effective ways of generating similar pictures.

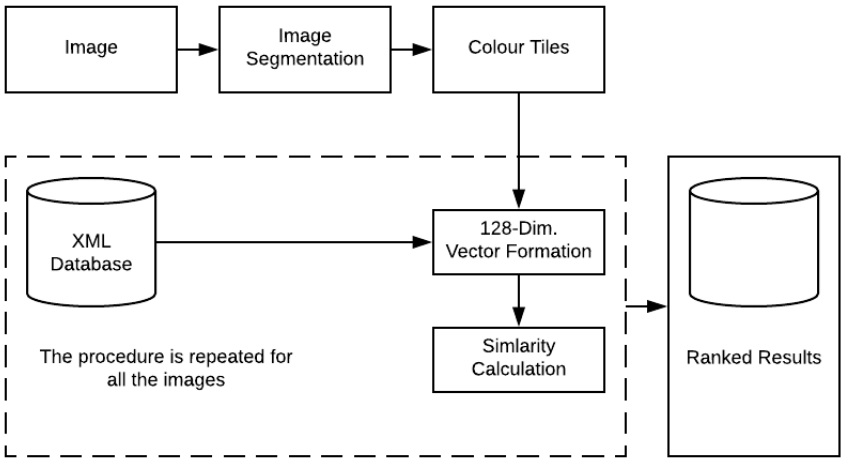


Figure 5.1: SpCD image retrieval procedure

We have experimented with different BCPGs and generated many pictures. For each BCPG, we selected eight pictures in its gallery. These sets of eight are given in Figures 5.2–5.8. Using the Img(Rummager) application, we calculated the mathematical similarity of the pictures in each set according to SpCD. For each set, we also calculated the average SpCD, in order to compare the similarity across different sets. In this chapter, rank represents the position of a picture with respect to similarity to a query picture. In this chapter, the rank ranges between 1 and 8, where the lower the rank, the higher the similarity of a picture to a given query picture. The query picture itself has rank equal to 1. In calculating this average, the query picture was left out, because it has SpCD value equal to 0 in all instances.

The average SpCD value was calculated as:

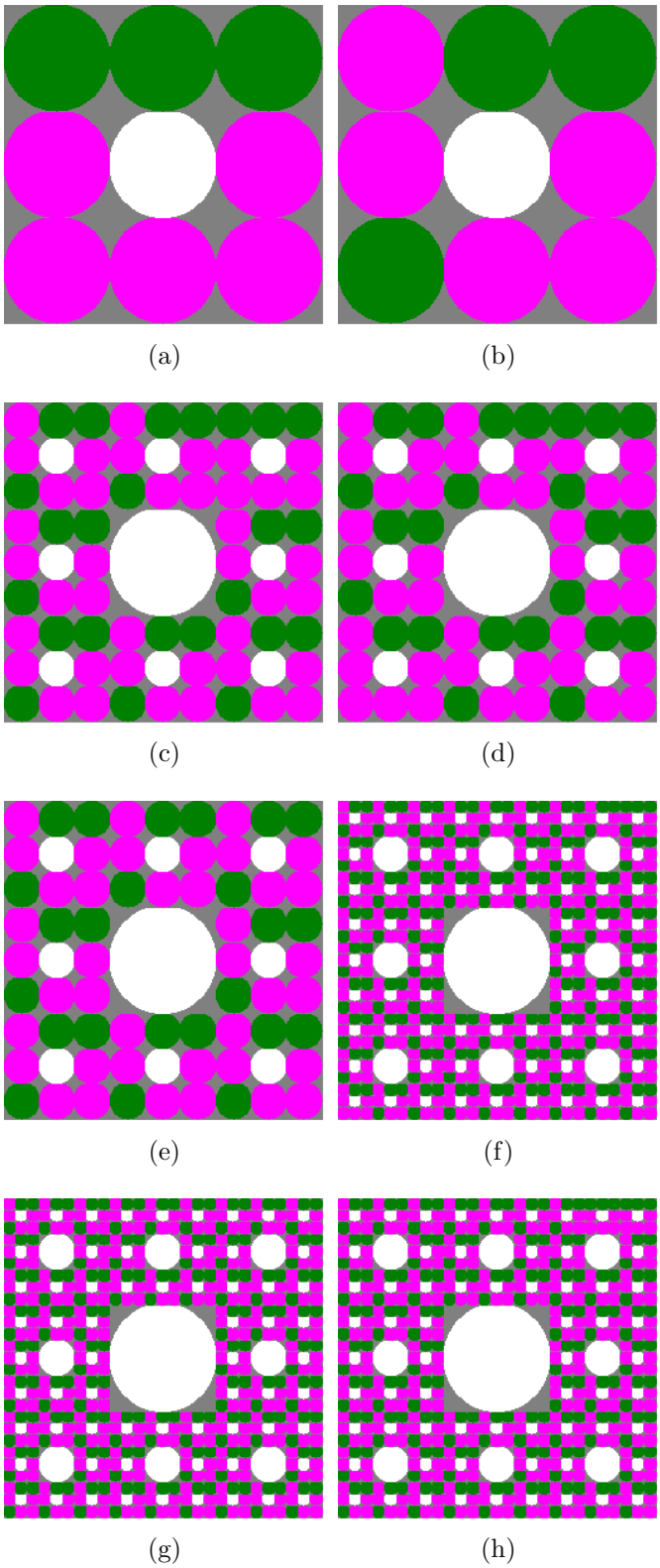
$$\text{Average SpCD} = \frac{\sum_{i=2}^{n} x_i}{n-1}$$

where

- n is the number of ranks, and
- x_i is the SpCD value of rank i

5.4 Results

Tables 5.1–5.10 present the results of the mathematical similarity measurements on the galleries in Figures 5.2–5.8. The SpCD values presented in Tables 5.1–5.10 are interpreted as follows: For the SpCD values, if the value is equal to 0, it means that the picture is identical to the query picture.



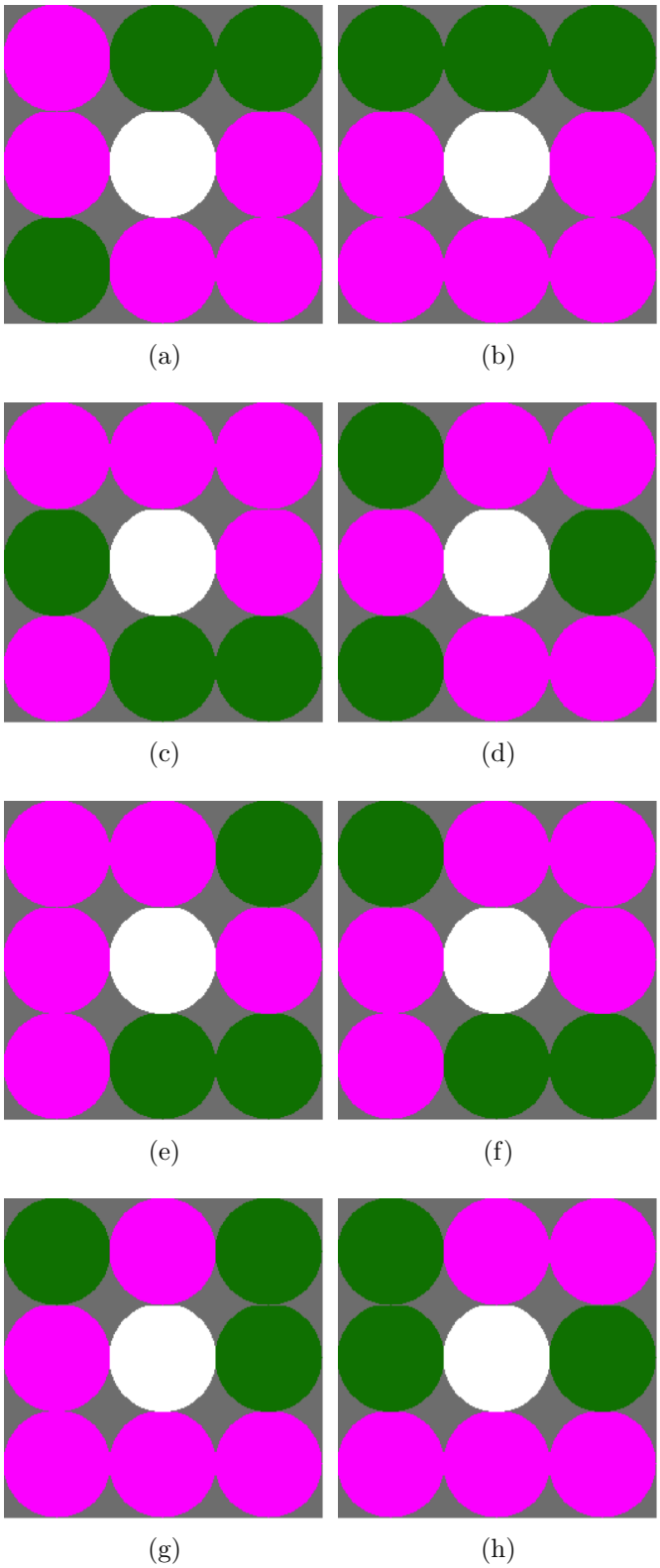
(a) (b)

(c) (d)

(e) (f)

(g) (h)

Figure 5.2: Sierpiński carpet, different refinements



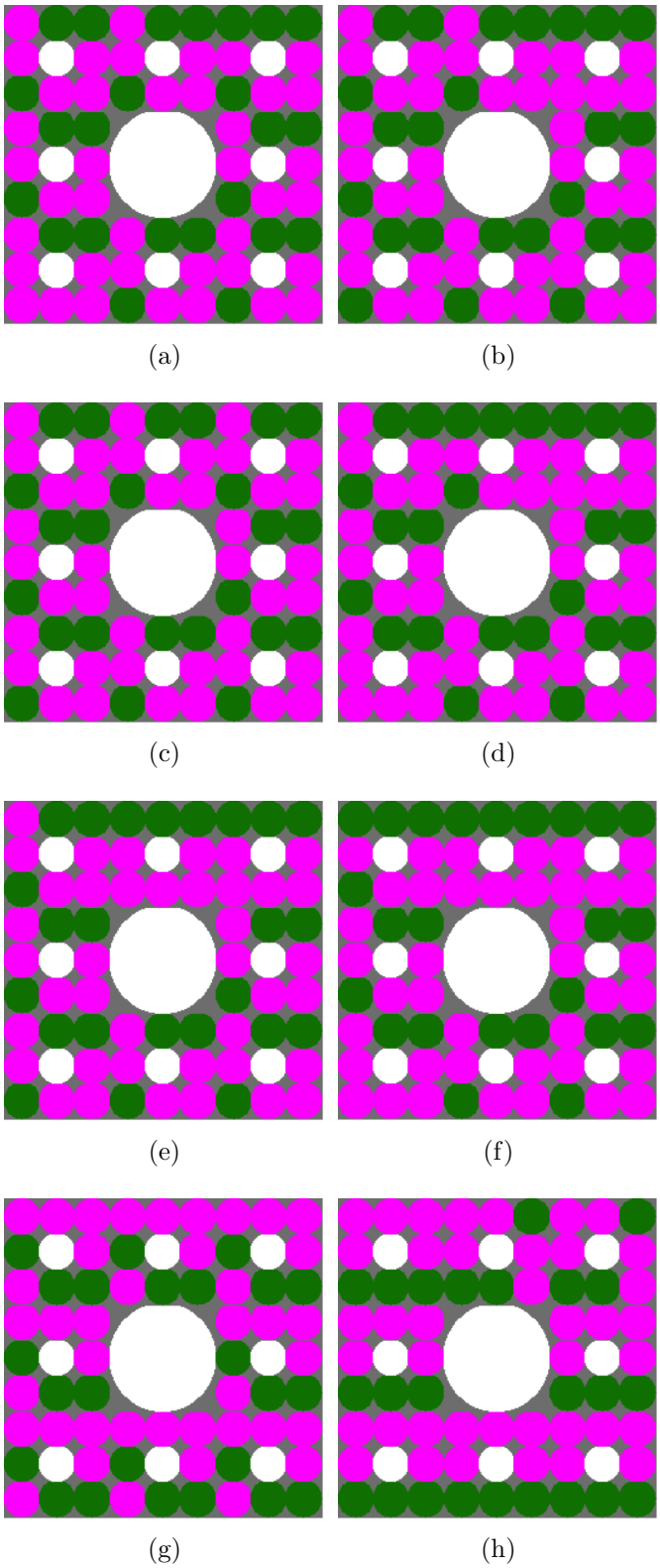
(a) (b)

(c) (d)

(e) (f)

(g) (h)

Figure 5.3: Sierpiński carpet, first refinement



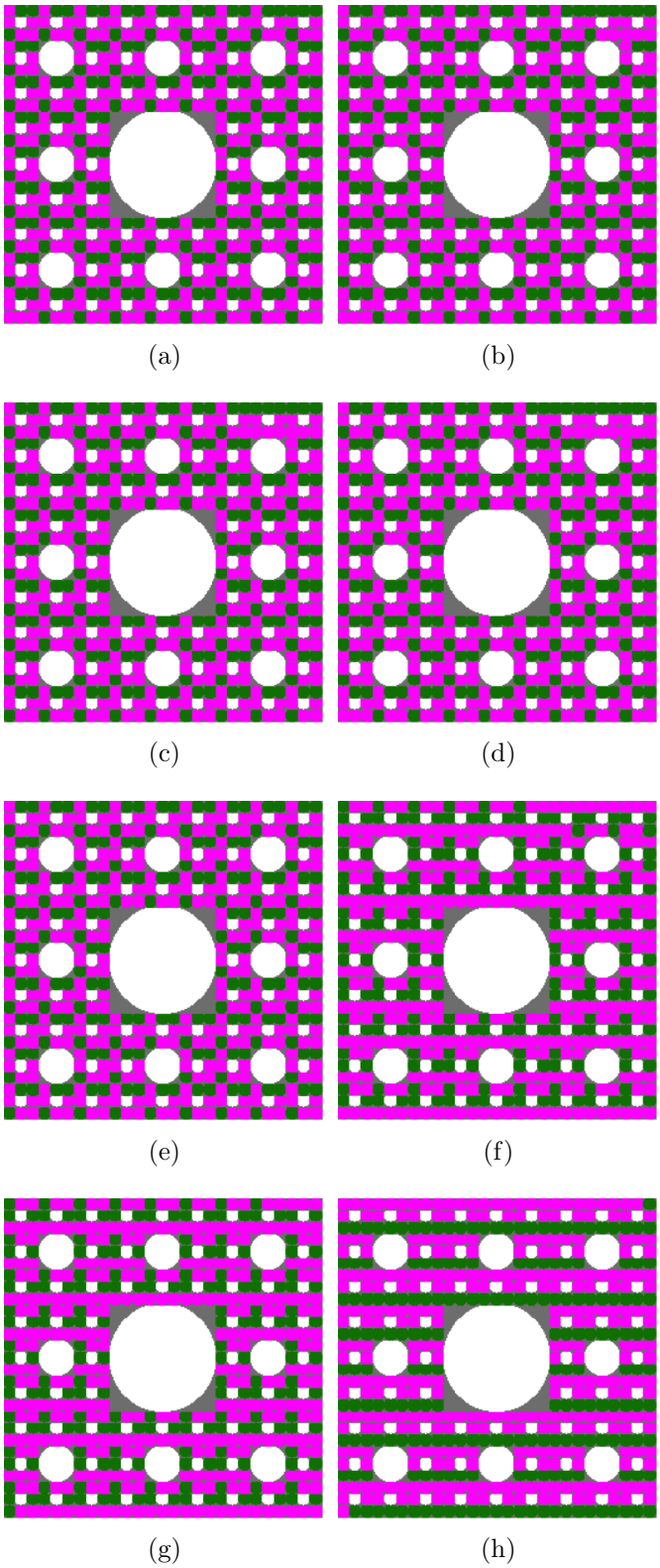
(a) (b)

(c) (d)

(e) (f)

(g) (h)

Figure 5.4: Sierpiński carpet, second refinement



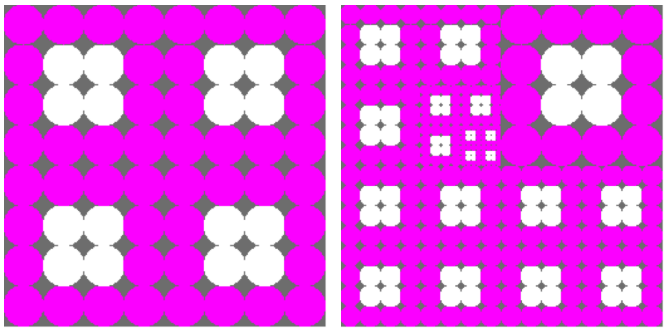
(a) (b)

(c) (d)

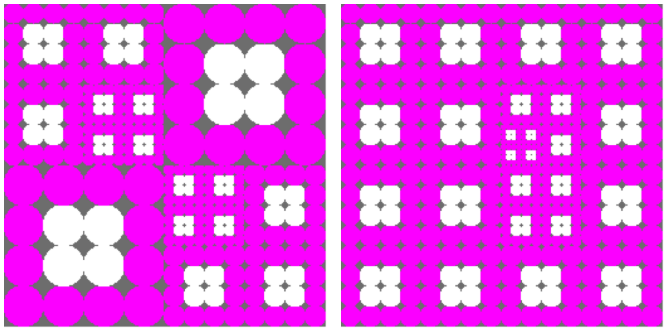
(e) (f)

(g) (h)

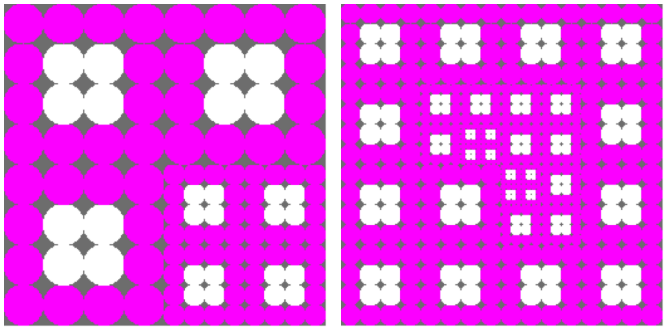
Figure 5.5: Sierpiński carpet, third refinement



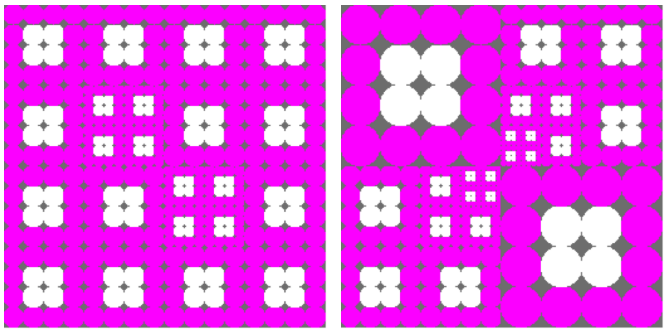
(a) (b)



(c) (d)

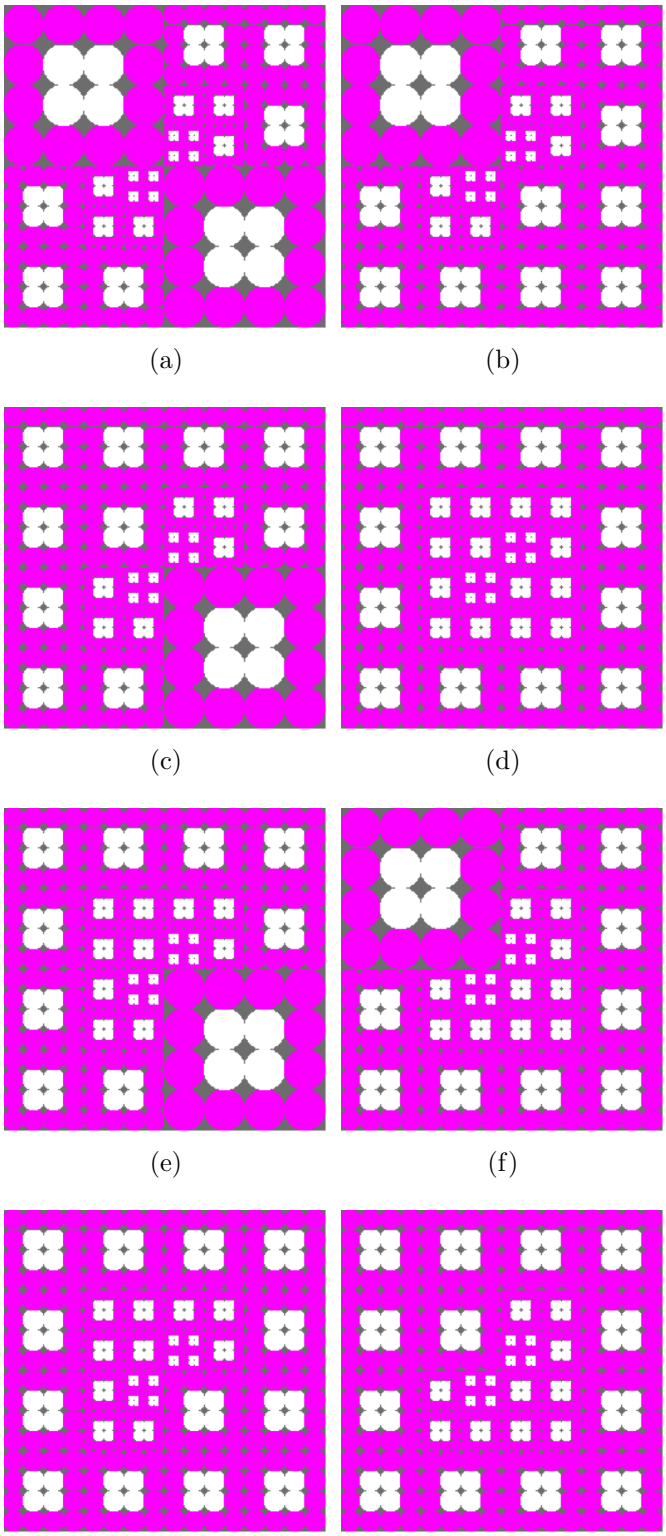


(e) (f)



(g) (h)

Figure 5.6: Sub-pictures refined up to three times



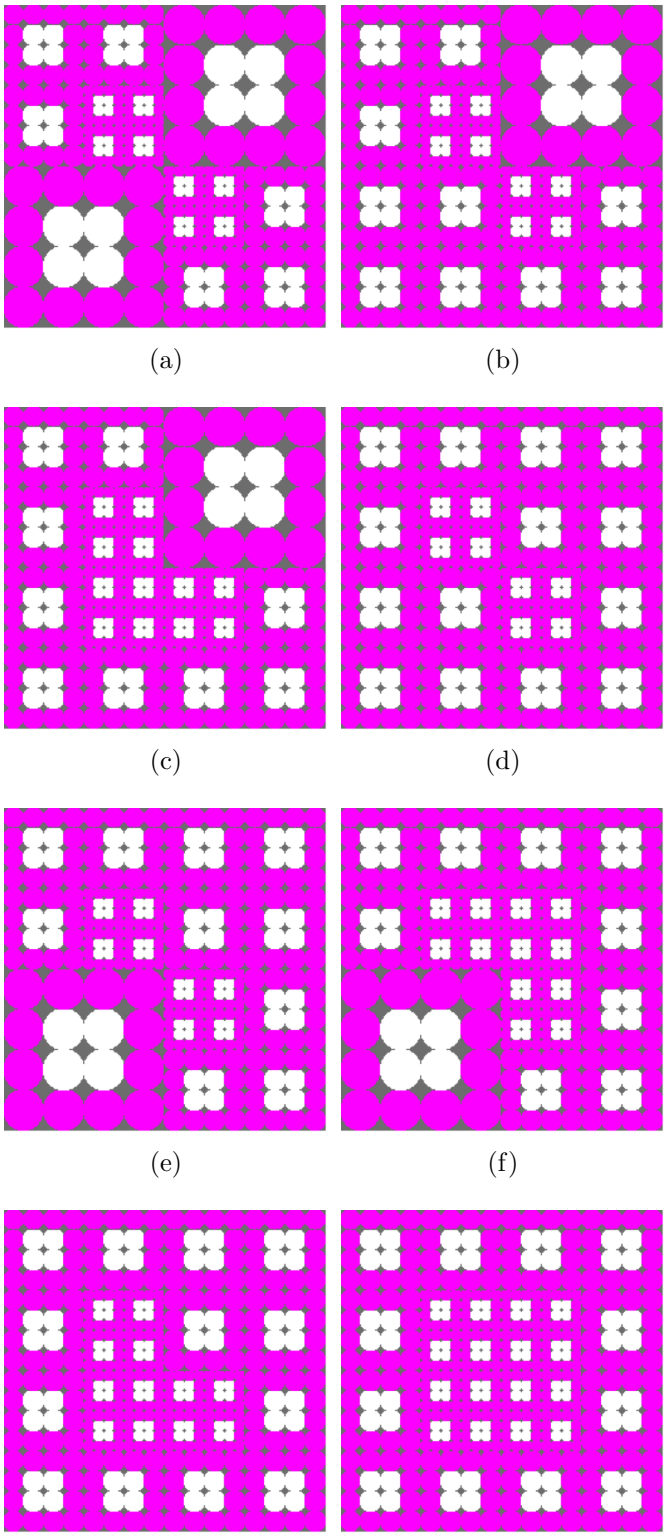
(a) (b)

(c) (d)

(e) (f)

(g) (h)

Figure 5.7: Top right quarter and the bottom left quarter refined exactly thrice



(a) (b)

(c) (d)

(e) (f)

(g) (h)

Figure 5.8: Top left quarter and the bottom right quarter subdivided exactly twice

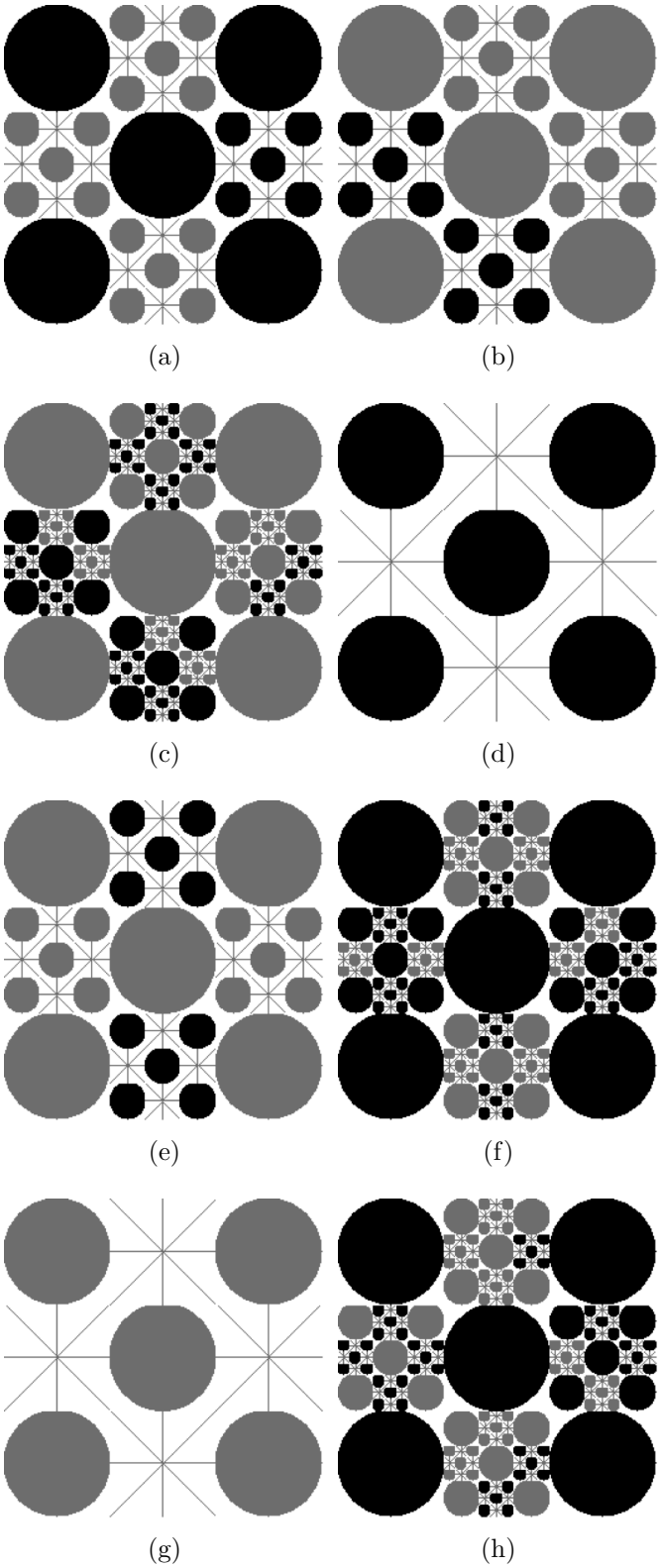


Figure 5.9: Different refinements with grey or black sub-pictures in the corners and centre

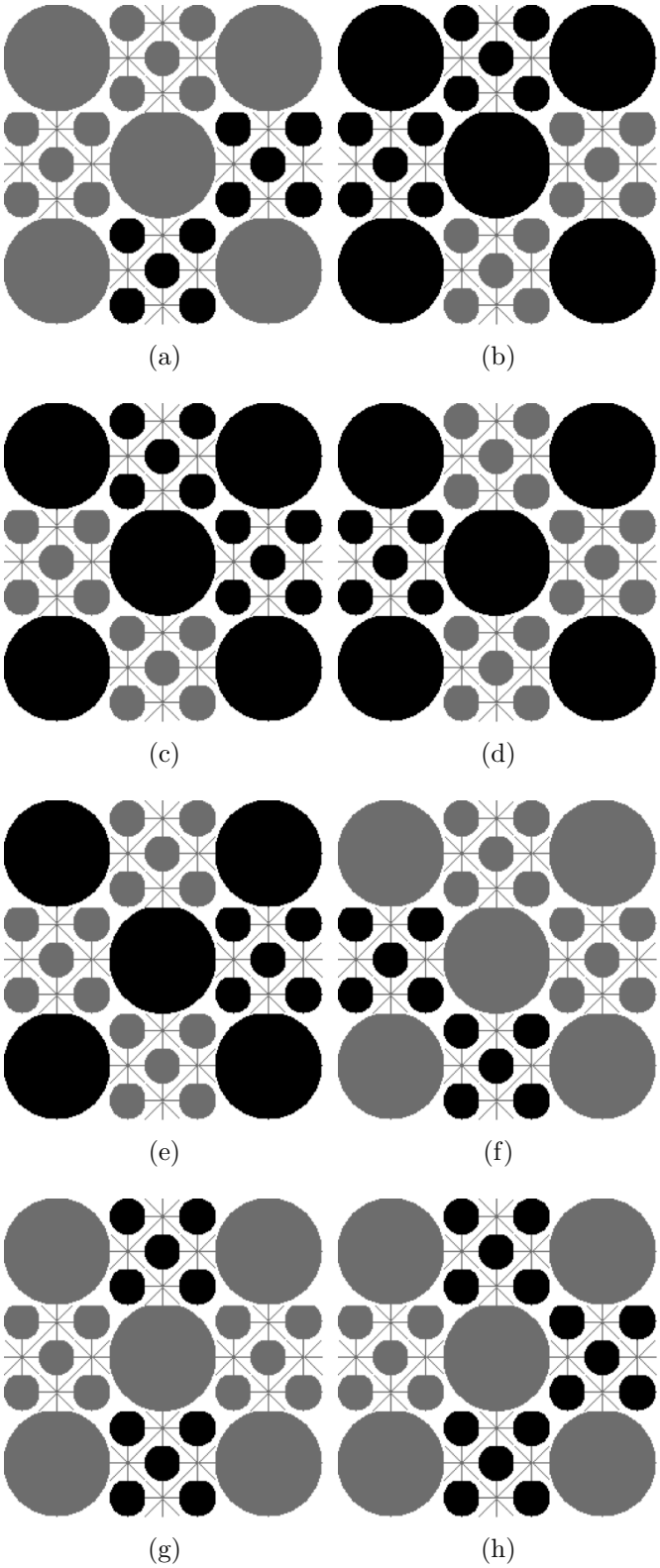


Figure 5.10: Second refinement with grey or black sub-pictures in the corners and centre

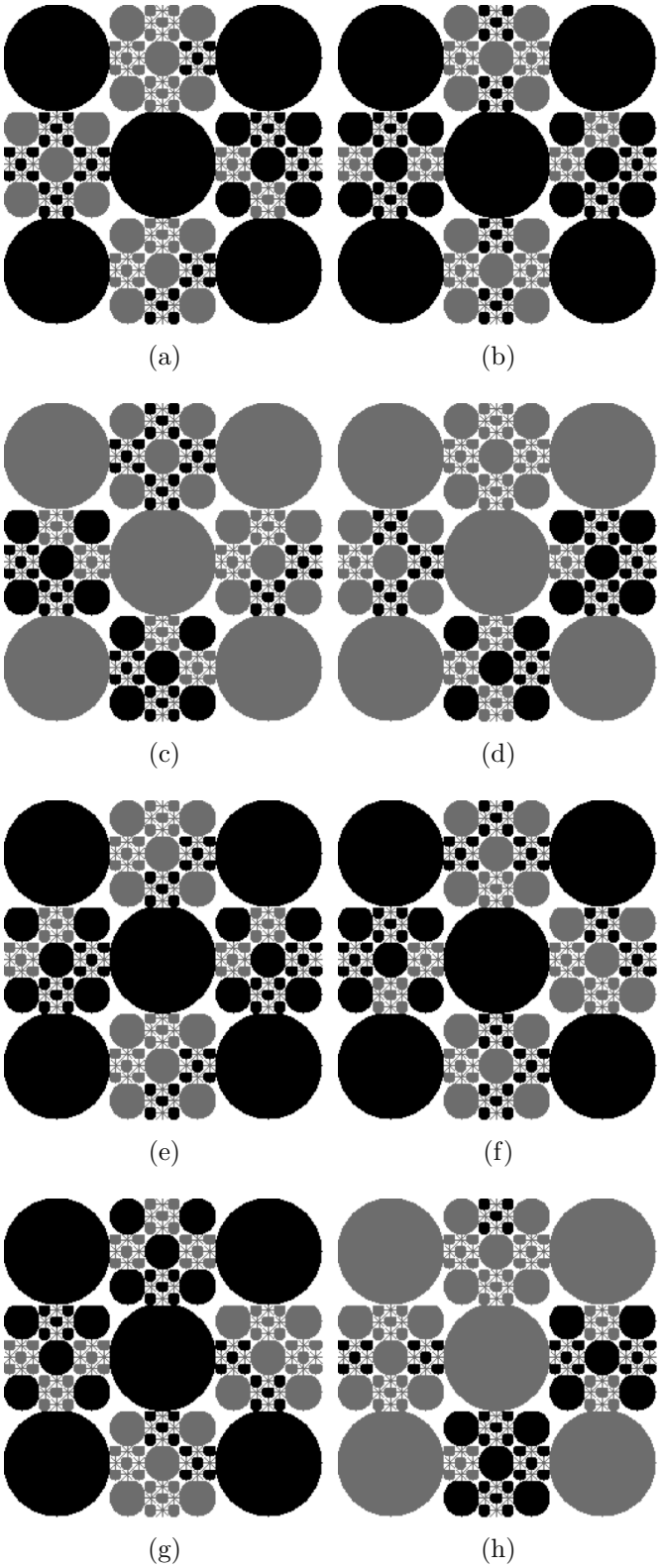


Figure 5.11: Third refinement with grey or black sub-pictures in the corners and centre

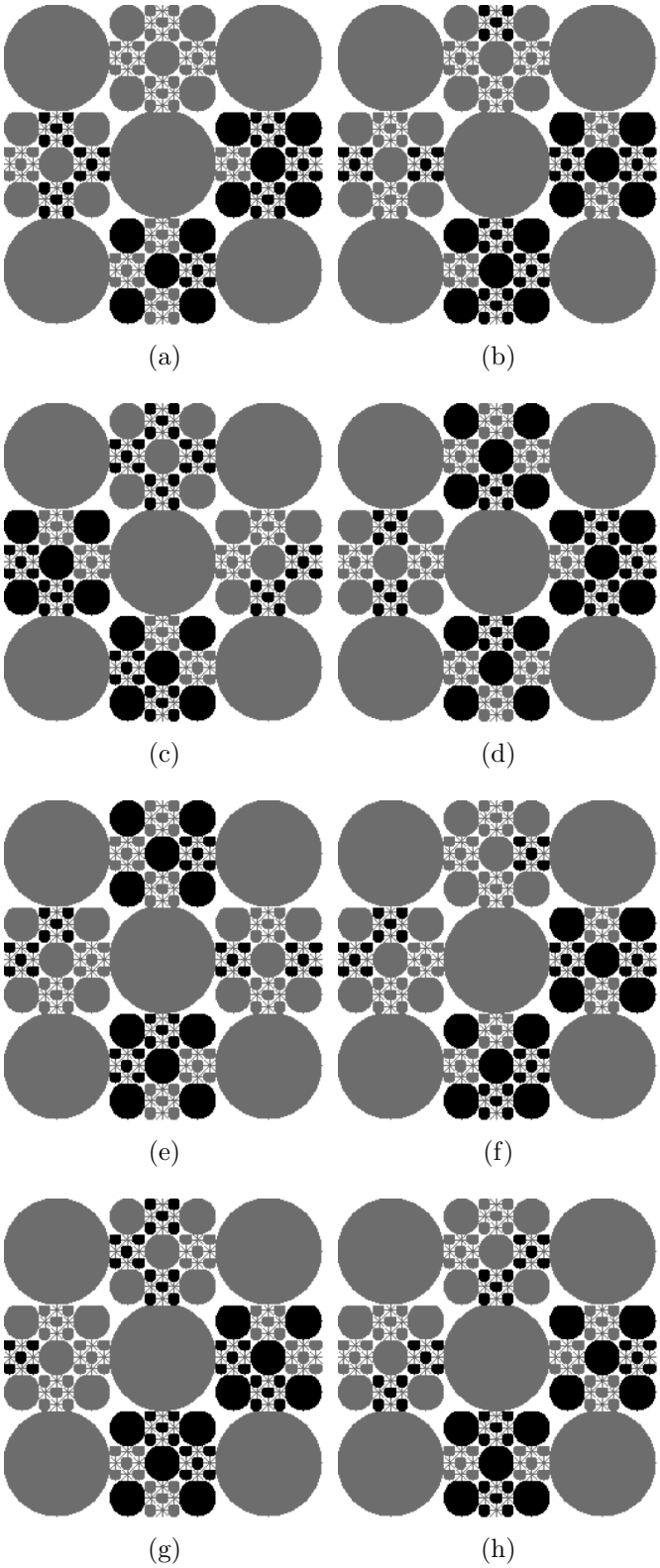


Figure 5.12: Third refinement with grey sub-pictures in the corners and centre

Moreover, it holds that the lower the SpCD value, the more similar the picture is to the query picture. In these tables, the picture with rank 1 is the query picture.

5.4.1 Limiting the Number of Refinements

Figures 5.2–5.5 consist of pictures that are in a specific level or levels of refinement. The colouring of the sub-pictures in the pictures of these galleries varies, thus creating pictures that are similar in terms of their level of refinement, but different in terms of colour placements. Figure 5.2 consists of pictures that are in the first, second and third levels of refinement. Figure 5.3 consists of pictures that are only in the first refinement. Figure 5.4 consists of pictures that are only in the second refinement and Figure 5.5 consists of pictures that are in the third refinement only.

Tables 5.1, 5.2, and 5.3 provide mathematical similarity results of Figures 5.2a, 5.2d and 5.2f to all the pictures in Figure 5.2 respectively. Consider the SpCD values for Figure 5.2a as query picture. The SpCD value of Table 5.1 shows that all the SpCD values are greater than 32, except for the query picture. This indicates that there is little similarity between the query picture, Figure 5.2a, and the other pictures in Figure 5.2. In Tables 5.2 and 5.3, we present the mathematical similarity for Figures 5.2d and 5.2f. In both cases, five of the seven values under consideration are less than 10, while the remaining two are greater than 32. Figure 5.2 contains pictures with three different levels of refinements. Moreover, the used query pictures are from three different levels of refinements. Figure 5.2a is in the first level of refinement, Figure 5.2d is in the second level of refinement and Figure 5.2f is in the third level of refinement. By examining their average SpCD values, we note that the higher the refinement of the query picture is, that is, the smaller its sub-pictures, the lower these SpCD values are.

Table 5.1: Similarity of Figure 5.2a to pictures in Figure 5.2

Picture	**Rank**	**SpCD value**
a	1	0
b	2	32.736
f	3	32.855
h	4	32.855
d	5	33.864
g	6	34.430
c	7	36.608
e	8	38.577
Average SpCD		34.560

Table 5.2: Similarity of Figure 5.2d to pictures in Figure 5.2

Picture	**Rank**	**SpCD value**
d	1	0
e	2	2.777
c	3	2.782
f	4	7.627
h	5	7.627
g	6	8.108
a	7	33.864
b	8	35.499
Average SpCD		14.040

Table 5.3: Similarity of Figure 5.2f to pictures in Figure 5.2

Picture	**Rank**	**SpCD value**
f	1	0
h	2	0
g	3	0.463
d	4	7.627
e	5	7.737
c	6	8.601
b	7	32.381
a	8	32.855
Average SpCD		12.809

Figure 5.3 consists only of the first level of refinement pictures of Figure 5.2. Table 5.4 presents similarity results of Figure 5.3a to all the pictures in Figure 5.3. All the SpCD values are greater than 33 and the average SpCD is 55.448, indicating little similarity between the query picture and the other pictures. This is probably a result of the fact that the pictures have large sub-pictures.

Table 5.4: Similarity of Figure 5.3a to pictures in Figure 5.3

Picture	**Rank**	**SpCD value**
a	1	0
b	2	33.120
g	3	49.099
d	4	50.000
e	5	51.541
h	6	66.321
c	7	67.652
f	8	70.406
Average SpCD		55.448

Figure 5.4 consists of only the second level of refinement pictures of Figure 5.2. Table 5.5 presents similarity results of Figure 5.4a compared to all the pictures in Figure 5.4. Five of the seven SpCD values in Table 5.5 are less than 5. The average SpCD value is 7.636, indicating Figure 5.4 has pictures that are more similar to the query picture than the pictures in Figure 5.2 and Figure 5.3.

Table 5.5: Similarity of Figure 5.4a to pictures in Figure 5.4

Picture	**Rank**	**SpCD value**
a	1	0
c	2	2.393
b	3	2.865
d	4	2.870
f	5	4.093
e	6	4.272
h	7	16.964
g	8	20
Average SpCD		7.636

Figure 5.5 consists of only the third level of refinement pictures of Figure 5.2. Table 5.6 shows similarity results of Figure 5.5a to all the pictures in Figure 5.5. The average SpCD value is 2.038, which is smaller

than those of Figures 5.2–5.4. This suggests that the pictures in Figure 5.5 are more similar to the query picture than the previously considered cases. Moreover, we note that the SpCD value for two pictures is zero, even though these pictures are not identical to the query picture.

Table 5.6: Similarity of Figure 5.5a to pictures in Figure 5.5

Picture	**Rank**	**SpCD value**
a	1	0
b	2	0
d	3	0
c	4	0.432
e	5	0.432
h	6	3.971
f	7	4.717
g	8	4.717
Average SpCD		2.038

5.4.2 Limiting Refinement of Sub-Pictures

Figures 5.6–5.8 consist of pictures with their sub-pictures refined to a given number of times. Figure 5.6 consists of pictures with the following structure: each quarter can be subdivided from zero to three times. Figure 5.7 consists of pictures with the top right quarter and the bottom left quarter subdivided exactly thrice, whereas the remaining quarters may be subdivided from zero to two times. Figure 5.8 consists of pictures with the top left quarter and the bottom right quarter subdivided exactly twice, while the remaining quarters may be subdivided from zero to two times.

Tables 5.7 and 5.8 present mathematical similarity results of Figures 5.6h and 5.6c to all the pictures in Figure 5.6 respectively. Consider the SpCD values for Figure 5.6h as a query picture. An average SpCD value of 2.494 was calculated. Consider the SpCD values for Figure 5.6c as a query picture. An average SpCD value of 1.145 was calculated indicating Figure 5.6c is more similar to all of the pictures in Figure 5.6 than Figure 5.6h.

Table 5.9 presents the results of the mathematical similarity of Figure 5.7a to all the pictures in Figure 5.7. An average SpCD of 0.686 was calculated, and the calculated value is smaller than the average SpCD value calculated in Tables 5.8 and 5.7. This implies that the pictures in Figure 5.7 are more similar to the query picture than those in Figure 5.6.

Table 5.10 presents the results of the mathematical similarity of Figure 5.8a to all the pictures in Figure 5.8. An average SpCD value of 0.902 was calculated and the value is smaller than the SpCD value calculated in

Table 5.7: Similarity of Figure 5.6h to pictures in Figure 5.6

Picture	**Rank**	**SpCD value**
h	1	0
a	2	1.253
e	3	1.524
f	4	1.857
g	5	1.857
d	6	2.468
b	7	2.735
c	8	5.764
Average SpCD		2.494

Table 5.8: Similarity of Figure 5.6c to pictures in Figure 5.6

Picture	**Rank**	**SpCD value**
c	1	0
f	2	0.624
g	3	0.624
b	4	0.893
e	5	0.906
a	6	1.257
d	7	1.276
h	8	2.439
Average SpCD		1.145

Table 5.9: Similarity of Figure 5.7a to pictures in Figure 5.7

Picture	**Rank**	**SpCD value**
a	1	0
c	2	0.286
b	3	0.309
e	4	0.606
f	5	0.607
h	6	0.864
g	7	0.935
d	8	1.197
Average SpCD		0.686

Table 5.7 and 5.8. Table 5.10 shows results which indicate that the pictures in Figure 5.8 are more similar to the query picture than the pictures in Figure 5.6.

Table 5.10: Similarity of Figure 5.8a to pictures in Figure 5.8

Picture	**Rank**	**SpCD value**
a	1	0
e	2	0
f	3	0.609
b	4	0.624
d	5	0.624
c	6	1.276
g	7	1.276
h	8	1.908
Average SpCD		0.902

5.4.3 Limiting Both Refinement Levels and Refinement of Sub-Pictures

Figures 5.9–5.12 consist of pictures that are in a specific level of refinement, and their sub-pictures are refined to a given number of times. The pictures are similar in terms of their level of refinement and refinement of sub-pictures but the placement of coloured objects varies. Figure 5.9 consists of pictures from the first, second and third level of refinements. For these pictures, the sub-pictures in the corners and centre have the same size in all levels of refinement. Figure 5.10 consists of pictures that are in the second level of refinement. Figures 5.11 and 5.12 consist of pictures that are in the third level of refinement. In Figure 5.11, the sub-pictures in the corners and centre are either black or grey, while in Figure 5.12 all the sub-pictures in the corners and centre are grey.

Table 5.11 presents the mathematical similarity results of Figure 5.9c to all the pictures in Figure 5.9. An average SpCD value of 17.565 was calculated.

Table 5.12 presents mathematical similarity results of Figure 5.10c to all the pictures in Figure 5.10. An average SpCD value of 11.805 was calculated. This indicates that the pictures in Figure 5.10 are more similar to the query picture than the pictures in Figure 5.9.

Table 5.13 provides the mathematical similarity results of Figure 5.11c to all the pictures in Figure 5.11. The calculated average SpCD for this gallery is 12.439. The result indicates that the pictures in Figure 5.11 are more similar to the query picture than the pictures in Figure 5.9. Moreover,

Table 5.11: Similarity of Figure 5.9c to pictures in Figure 5.9

Picture	**Rank**	**SpCD value**
c	1	0
b	2	8.227
e	3	9.937
f	4	15.770
a	5	16.250
e	6	17.302
g	7	27.700
d	8	27.771
Average SpCD		17.565

Table 5.12: Similarity of Figure 5.10c to pictures in Figure 5.10

Picture	**Rank**	**SpCD value**
c	1	0
e	2	1.719
b	3	3.418
d	4	5.070
h	5	16.408
a	6	17.890
f	7	18.892
g	8	19.244
Average SpCD		11.805

we note that the average SpCD in Table 5.13 is not very different from that calculated in Table 5.12. This is probably a result of the fact that majority of the sub-pictures in the pictures in these galleries are of the same the size and shape.

Table 5.13: Similarity of Figure 5.11c to pictures in Figure 5.11

Picture	**Rank**	**SpCD value**
c	1	0
d	2	3.999
h	3	5.921
f	4	13.703
g	5	15.186
a	6	15.770
b	7	16.250
e	8	16.250
Average SpCD		12.439

Table 5.14 presents the mathematical similarity results of Figure 5.12c to all the pictures in Figure 5.12. We calculated an average SpCD value of 3.997, which is smaller than that calculated in Tables 5.11, 5.12 and 5.13 indicating Figure 5.12 pictures are more similar to the query picture than those of Figures 5.9, 5.10 and 5.11. This may be a result of the same colouring for the sub-pictures in the corners and centre in all the pictures in Figures 5.12.

Table 5.14: Similarity of Figure 5.12c to pictures in Figure 5.12

Picture	**Rank**	**SpCD value**
c	1	0
d	2	2.040
h	3	2.702
e	4	3.355
a	5	3.999
f	6	4.666
g	7	5.298
b	8	5.921
Average SpCD		3.997

5.5 Discussion

The galleries were grouped into two categories in order to test different methods of generating structurally similar pictures and investigate which methods are effective in generating mathematically similar pictures. The first method was to limit the number of refinements in a picture. The corresponding galleries are given in Figures 5.2–5.5. Figure 5.2 consists of pictures in three different levels of refinement, while the galleries given in Figures 5.3–5.5 consists of pictures that are in the same level of refinement. The mathematical similarity of the pictures in Figures 5.4 and 5.5 is higher than that of the pictures in Figure 5.2, regardless of the query picture used for the latter.

However, the opposite holds for Figure 5.3. Based on the results, we can state that BCPGs and RCPGs are able to generate mathematically similar pictures but, if the grammar is modified to generate pictures that are in the same level of refinement to the query picture, there is an excellent chance of improving the similarity of the pictures. Moreover, in the galleries consisting of pictures that are on the same level of refinement, the similarity increased with the level of refinement to some extent. The similarity of Figure 5.3 was not better than that of Figure 5.2. This was probably because the sub-pictures in Figure 5.3 were large compared to most in Figure 5.2 and the placement of colours from one picture to the next differed a good deal. According to its definition, SpCD measures the similarity of pictures based on the spatial colour distribution. We observe that any significant colour shift leads to a big difference in similarity values. For example, consider the similarity values of Figure 5.4g and Figure 5.4h in Table 5.5. They are much higher than the other values in Table 5.5, even though all pictures in this gallery have sub-pictures of the same size.

The second method was to limit the refinement of sub-pictures. The corresponding galleries are given in Figures 5.6–5.8. The pictures in Figures 5.7 and 5.8 have a higher mathematical similarity than the pictures in Figure 5.6 regardless of the query picture used for the latter. This implies that the mathematical similarity of the pictures improves if all the pictures in the gallery have the same sub-pictures in common.

The third method was to limit both refinement levels and refinement of sub-pictures. The corresponding galleries are given in Figures 5.9–5.12. Figures 5.10, 5.11 and 5.12 have a higher mathematical similarity than that of the pictures in Figure 5.9. This indicates that, if the grammar is modified to generate pictures that are in the same level of refinement as the query picture, there is an excellent chance of improving the similarity of the pictures. Figure 5.12 has a higher mathematical similarity than any of Figures 5.9–5.11. This implies that the mathematical similarity of the

pictures improves if all the pictures in the gallery have the same sub-pictures in common as well as being in the same level of refinement.

Moreover, the pictures generated by limiting the refinement of sub-pictures have lower SpCD values than most of the pictures where we limited the number of refinements or limited both refinement levels and refinement of sub-pictures. Presumably this is because fewer colours were used in the former galleries than in the latter.

All methods used to modify a given grammar to generate structurally similar pictures have shown to be effective in generating mathematically similar pictures. Judging by the average SpCD values for the pictures generated by the three methods, the second method produced the better results.

Since these methods used galleries that have different properties, we cannot make a conclusion as to which method is better based on these results. We can only note that the described methods produced galleries with a higher mathematical similarity than the original galleries. Moreover, we note that one can modify a grammar to generate a finite number of relevant pictures.

From the results, we learn that, when sub-pictures are too refined, SpCD sometimes fails to tell the difference between pictures. On the other hand, pictures with large sub-pictures are easy to differentiate and tend to be less similar. SpCD found these pictures to be dissimilar (for example, Figure 5.3, see Table 5.4).

5.6 Conclusion

This chapter presented similar pictures generated by BCPGs and RCPGs and the similarity results calculated by the SpCD similarity measure to determine the most effective strategies of generating similar pictures using BCPGs and RCPGs. Both BCPGs and RCPGs can be written and modified in many ways in order to generate similar pictures. Three different strategies were used to produce the pictures tested in this chapter. The presented methods are limiting the number of refinements of the generated pictures, limiting the refinement of some sub-pictures in the pictures and limiting both refinement levels and refinement of sub-pictures. All the strategies presented showed that, when the grammar was modified to generate structurally similar pictures, the similarity of the generated pictures improved.

The results of this research imply that both BCPGs and RCPGs can be used to generate similar pictures by selecting a picture and modifying its grammar to generate only pictures that are similar to it. We also observe that using the methods presented here can improve the similarity of the generated pictures. As different galleries are used for each method, we cannot

make a general conclusion as to which of the methods is better. Moreover, we acknowledge that modifying grammars to produce structurally similar pictures improves the similarity of the generated pictures.

Chapter 6

Perceptual similarity

6.1 Introduction

In this chapter we present the results on how humans perceive the similarity (perceptual similarity) of pictures generated by bag context picture grammars and random context picture grammars. This chapter contains the work that was presented at SIMULTECH, 2018 [45].

Determining picture similarity is a crucial element in many applications that require the comparison of pictures on different aspects, like colour, texture, layout and theme. Applications such as search engines, picture database retrieval systems, picture generators and visual password schemes are some of the applications that may require determining the degree of similarity of pictures [34, 63].

Determining picture similarity is very important for our research as we focus on generating structurally similar pictures using bag context picture grammars and random context picture grammars. An end goal of our work is in generating visual passwords and appropriate distractors (pictures which are similar to the password picture) for a visual password system and it is, thus, necessary to evaluate if the generated pictures are similar. As visual password systems are used by humans, it is important to determine how humans perceive the similarity of the generated pictures, and determine if the chosen similarity measure is consistent with the human view of similarity.

In this chapter, we analyse how humans perceive the similarity of these generated pictures. We conducted an online survey to determine how humans determine the similarity of syntactically generated pictures and to determine if the human view of similarity is consistent with the selected mathematical similarity measure, the spatial colour distribution descriptor [6]. The results of the online survey are then compared with the results of applying the SpCD to the same pictures. We used discounted cumulative gain (DCG) to evaluate the consistency of ranking of perceptual similarity and the SpCD.

6.2 Results

For this research, it is important to measure if perceptual similarity correlates to the SpCD, because we need to be sure that the results from the mathematical measure reflect what people think. For this, we conducted an online survey to evaluate the level of consistency between perceptual similarity and the SpCD.

We obtained 408 responses through the online survey. Most of the respondents were staff members or students from University of the Witwatersrand. Other respondents were contacts of the authors. All the galleries used in the survey were generated by the grammars in Chapter 4. Examples 4.1–4.7 discuss the generation of pictures for the galleries in Figures 6.1–6.7 respectively.

The survey contained the following points:

Ranking of pictures in gallery: We showed the respondents the galleries in Figures 6.1–6.7. For each gallery a respondent had to rank the pictures in the gallery in terms of how similar they felt the picture was to a given picture, in particular the picture with label (c) in each gallery. In the ranking, the value 1 was given to the most similar picture and 5 to the least similar picture. The picture (c) was used both as the query picture and as a picture in the gallery to check for outliers.

Similarity of pictures in gallery: For Figures 6.1–6.7, we asked respondents to select the statement that best described the similarity of the pictures in that gallery. The statements were:

- not at all similar,
- somehow similar,
- similar,
- very similar, and
- identical.

Ranking of galleries: For Figures 6.1–6.4 and Figures 6.5–6.7, respectively, we asked respondents to rank the galleries from the gallery with the pictures that are most similar to each other to the gallery with the pictures that are least similar to each other.

Factors that determine similarity: We asked respondents which factor they considered the most important when determining the similarity of the pictures in a gallery. We provided them with the following options:

- colours present in the picture,

- distribution of the colours in the picture,
- objects in the picture,
- distribution of the objects in the picture, and
- other (specify).

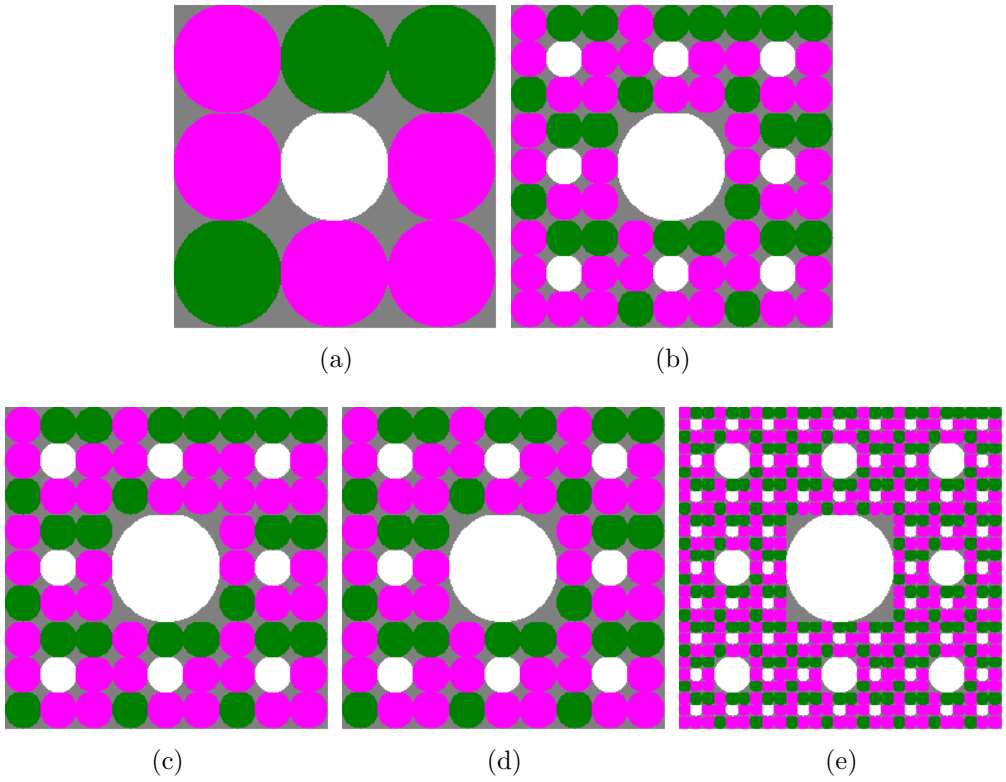


Figure 6.1: Gallery A: Sierpiński carpet, different refinements

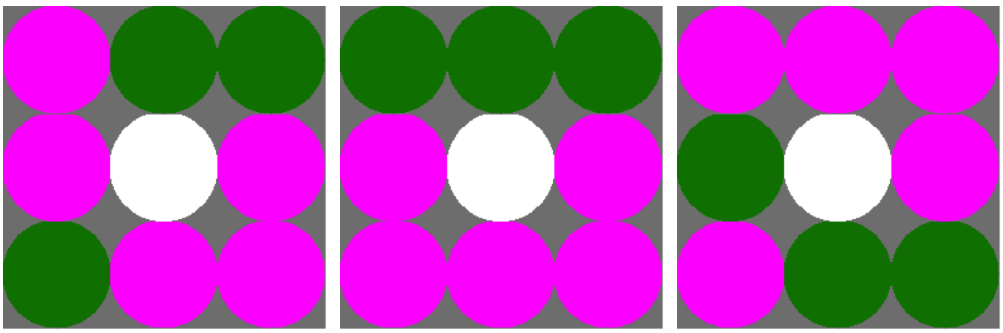
6.2.1 Ranking of Pictures in Gallery

As stated above, picture (c) was used as the query picture in each gallery, that is, all pictures were compared to picture (c) to determine their similarity to it.

The results of the SpCD and the online survey are presented in Tables 6.1–6.7. Each table is structured as follows:

Rank: The first column presents the picture ranking from 1 (most similar) to 5 (least similar).

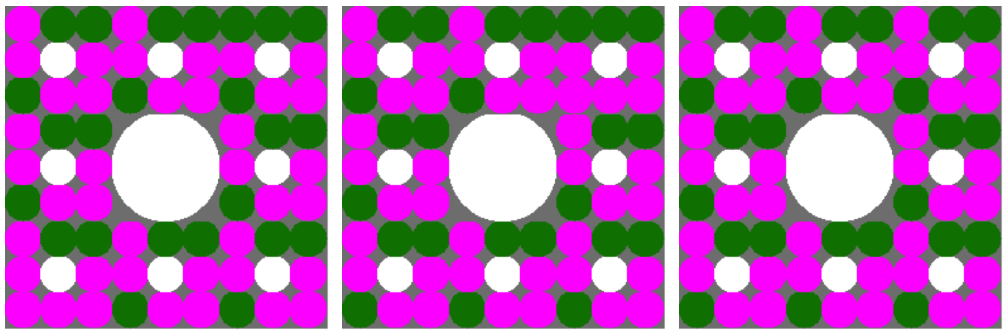
SpCD: The second column presents the SpCD. It is divided into two columns, the first giving the picture label and the second its SpCD value.



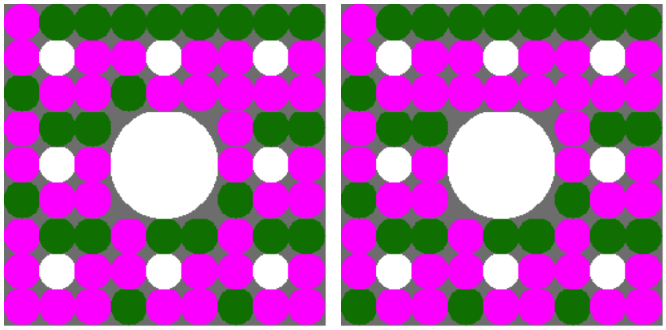
(a) (b) (c)

(d) (e)

Figure 6.2: Gallery B: Sierpiński carpet, first refinement

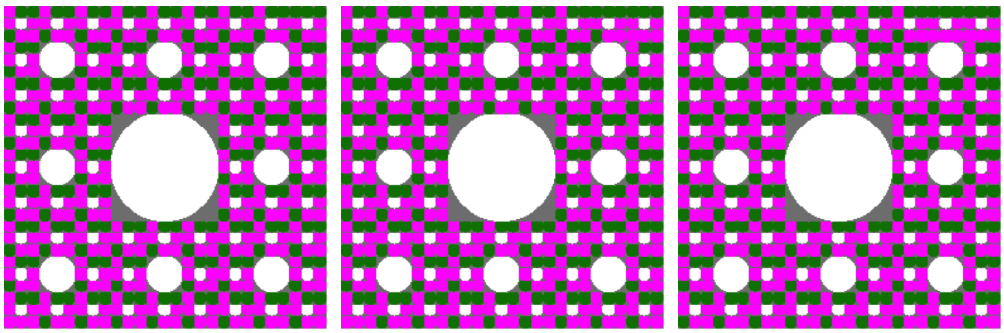


(a) (b) (c)

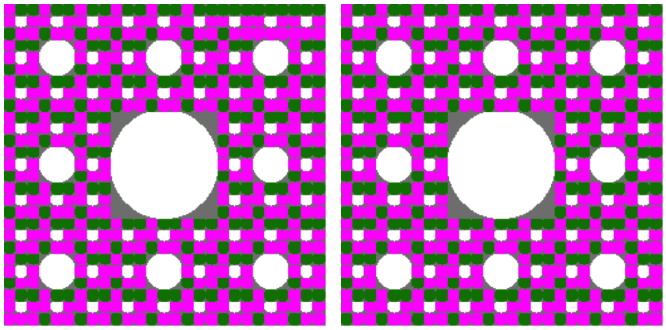


(d) (e)

Figure 6.3: Gallery C: Sierpiński carpet, second refinement

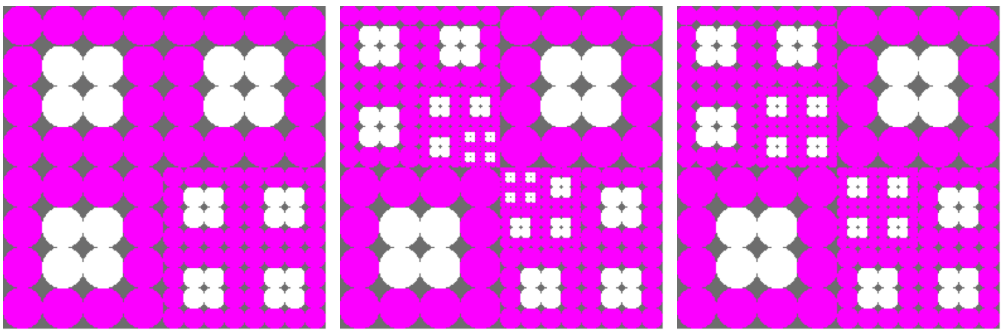


(a) (b) (c)

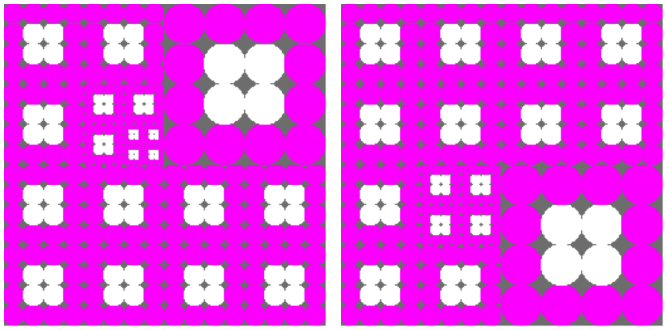


(d) (e)

Figure 6.4: Gallery D: Sierpiński carpet, third refinement

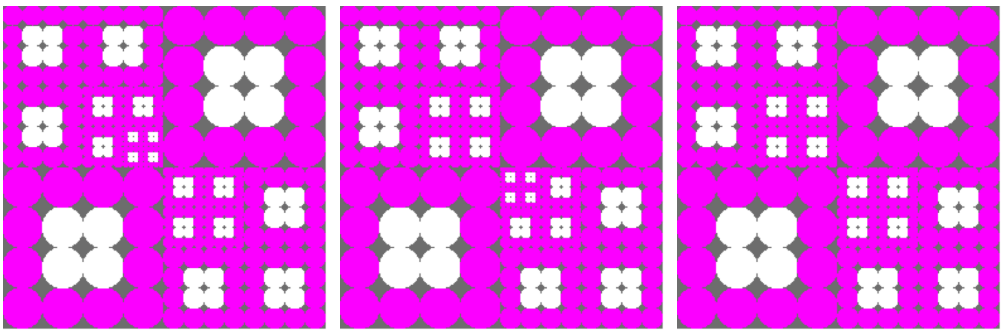


(a) (b) (c)

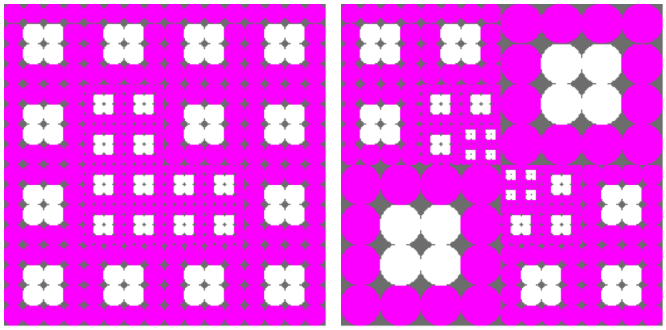


(d) (e)

Figure 6.5: Gallery E: Flowers

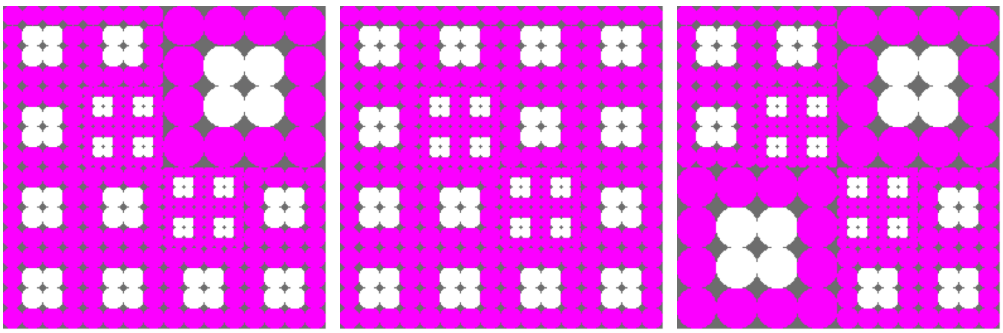


(a) (b) (c)

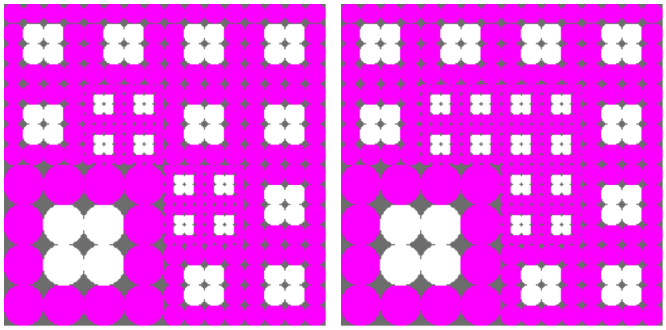


(d) (e)

Figure 6.6: Gallery F: Flowers



(a) (b) (c)



(d) (e)

Figure 6.7: Gallery G: Flowers

Perceptual: The third column presents the perceptual similarity. It is divided into two columns, the first giving the picture label and the second its average perceptual ranking.

The average perceptual ranking (or score) AV over all the respondents was calculated as:

$$\text{AV} = \frac{\sum_i^n w_i x_i}{\sum_i^n x_i} \ ,$$

where

- n is the number of ranks,
- w_i is the weight of the rank, where the picture that was ranked as the most similar is given the weight of 5 and the least similar picture is given the weight of 1, and
- x_i is the number of responses for each possible answer.

For the ranking according to the SpCD, the picture with the smallest value is the most similar to the query picture, while the picture with the largest value is the least similar. On the other hand, for the ranking according to the perceptual similarity, the picture with the largest value is the most similar to the query picture and the picture with the smallest value the least similar.

The SpCD values and the average perceptual ranking cannot be compared directly, because they use different unit measures. We therefore compare the ranking of the pictures by the two measures.

In the following, each of Tables 6.1–6.7 is discussed briefly.

Consider Table 6.1, which gives the results for Gallery A in Figure 6.1. For this gallery, the SpCD is, to a degree, consistent with human perceptual similarity as three of the five pictures were ranked the same for both similarity measures.

Table 6.1: Similarity of Figure 6.1c to pictures in Figure 6.1

	SpCD		Perceptual	
Rank	**Image**	**Value**	**Image**	**Score**
1	c	0	c	4.53
2	d	1.932	b	3.80
3	b	2.782	d	3.31
4	e	8.601	e	1.89
5	a	33.425	a	1.62

Consider Table 6.2, which gives the results for Gallery B in Figure 6.2. For this gallery, the SpCD is, to a degree, consistent with human perceptual

similarity. Three of the five pictures were ranked at the same positions. There is a small score difference of 0.17 in the perceptual similarity of the remaining two pictures, implying that respondents found these pictures to be very similar.

Table 6.2: Similarity of Figure 6.2c to pictures in Figure 6.2

	SpCD		Perceptual	
Rank	**Image**	**Value**	**Image**	**Score**
1	c	0	c	4.79
2	d	23.437	e	3.00
3	e	30.126	d	2.83
4	a	67.653	a	2.79
5	b	73.297	b	1.59

Consider Table 6.3, which gives the results for Gallery C in Figure 6.3. For this gallery, the SpCD is to a degree consistent with human perceptual similarity. Both measures ranked the first picture on the same position. However, there is a difference at Ranks 2 and 3. Both measures place pictures (a) and (b) at these ranks but in different orders. However, the difference of the weighted score of 0.21 suggests that respondents found these pictures to be very similar. There is a similar situation at Ranks 4 and 5. Both measures place pictures (d) and (e) at these ranks but in different orders. Moreover, in this case, the difference of the weighted score of 0.04 suggests that respondents found these pictures to be very similar.

Table 6.3: Similarity of Figure 6.3c to pictures in Figure 6.3

	SpCD		Perceptual	
Rank	**Image**	**Value**	**Image**	**Score**
1	c	0	c	4.67
2	a	2.394	b	3.51
3	b	3.426	a	3.30
4	e	4.830	d	1.83
5	d	5.291	e	1.79

Consider Table 6.4, which gives the results for Gallery D in Figure 6.4. It shows no correlation between the two measures. In fact, the ranking of the pictures in the online survey suggests that the respondents were not able to tell the difference between the pictures. For example, the respondents ranked Figure 6.4c, which is the query picture, third instead of first. This might be because the pictures in this gallery have very small sub-pictures, which might have made it difficult for respondents to distinguish one picture from

another. It is worth noting that the SpCD values for this gallery are very small, and that the difference between values in Table 6.4 is small compared to that in Tables 6.1, 6.2 and 6.3. Moreover, we observe that the SpCD could not measure the difference between pictures which are different. For example, it ranked pictures (c) and (e) as identical, and similarly pictures (a), (b) and (d). We assume the underlying reason is that the SpCD cannot measure the difference between pictures that have such small sub-pictures.

Table 6.4: Similarity of Figure 6.4c to pictures in Figure 6.4

	SpCD		Perceptual	
Rank	**Image**	**Value**	**Image**	**Score**
1	c	0	d	4.13
2	e	0	b	3.76
3	a	0.433	c	3.29
4	b	0.433	a	2.62
5	d	0.433	e	1.39

Consider Table 6.5, which gives the results for Gallery E in Figure 6.5. For this gallery, the SpCD is to a degree, consistent with human perceptual similarity. Three pictures were ranked the same by both measures. The measures differed at Ranks 3 and 4. However, the difference of the weighted score of 0.27 suggests that respondents found these pictures to be very similar.

Table 6.5: Similarity of Figure 6.5c to pictures in Figure 6.5

	SpCD		Perceptual	
Rank	**Image**	**Value**	**Image**	**Score**
1	c	0	c	4.68
2	b	0	b	3.97
3	d	0.894	a	2.51
4	a	0.907	d	2.24
5	e	1.865	e	1.63

Consider Table 6.6, which gives the results for Gallery F in Figure 6.6. In this case, the SpCD is consistent with human perceptual similarity as both measures ranked the pictures in the same order.

Consider Table 6.7, which gives the results for Gallery G in Figure 6.7. For this gallery, the SpCD is to a degree, consistent with human perceptual similarity. Two pictures, namely pictures (c) and (b), were ranked the same by both measures. Moreover, both measures ranked picture (d) higher than picture (e). However, human perceptual similarity ranked picture (a) higher

Table 6.6: Similarity of Figure 6.6c to pictures in Figure 6.6

	SpCD		Perceptual	
Rank	**Image**	**Value**	**Image**	**Score**
1	c	0	c	4.63
2	a	0	a	3.48
3	b	0	b	3.31
4	e	0	e	2.44
5	d	1.276	d	1.31

than pictures (d) and (e) whereas the SpCD ranked picture (a) lower than pictures (d) and (e). Moreover, the SpCD assigned pictures (a) and (b) the same values, whereas perceptual similarity did not consider these pictures to be identical.

Table 6.7: Similarity of Figure 6.7c to pictures in Figure 6.7

	SpCD		Perceptual	
Rank	**Image**	**Value**	**Image**	**Score**
1	c	0	c	4.76
2	d	0	a	3.46
3	e	0.610	d	2.95
4	a	0.625	e	1.97
5	b	0.625	b	1.81

6.2.2 Similarity of Pictures in Gallery

In the second section of the survey, respondents were shown Figures 6.1–6.7 and asked to select the statement that best described the similarity of the pictures within that gallery. The options were:

- not at all similar,
- somehow similar,
- similar,
- very similar, or
- identical.

Consider Table 6.8, which shows how humans evaluated the similarity of pictures within each gallery. The value 1 indicates that the pictures are not at all similar, while 5 indicates that the pictures are identical. All the values

in Table 6.8 are higher than 1, which implies that the respondents found the pictures to be similar to some degree. The highest values are for Galleries C and D in Figures 6.3 and 6.4, which implies that the respondents considered these galleries to have the pictures that are most similar to each other.

Table 6.8: Perception of similarity of pictures in each gallery

Rank	**Gallery**	**Perceptual value**
1	A	1.82
2	B	2.40
3	C	3.12
4	D	3.64
5	E	2.17
6	F	2.66
7	G	2.24

6.2.3 Ranking of Galleries

In the third section of the survey, respondents were asked to rank the galleries in Figures 6.1–6.4 and Figures 6.5–6.7, respectively, from the gallery containing pictures that are most similar to each other to the gallery containing pictures that are least similar to each other.

Consider Table 6.9, which gives the results for Figures 6.1–6.4. Humans ranked Gallery D in Figure 6.4 highest, that is, as the gallery with pictures that are most similar to each other. This view correlates with the SpCD measures for this gallery (Table 6.4), which are very low (0 or 0.433), implying that the pictures are very similar to the query picture and each other. Humans ranked Gallery C in Figure 6.3 second. This view correlates with the SpCD measures for this gallery (Table 6.3), which are the second lowest for the four galleries under consideration. Humans ranked Gallery A in Figure 6.1 last, that is, as the gallery with pictures that are least similar to each other. This does not correlate with the SpCD measures for the four galleries. The SpCD values are the highest for Gallery B in Figure 6.2. Moreover, they differ a great deal from one picture to another, implying that Figure 6.2 is the gallery containing the least similar pictures. A possible explanation for this discrepancy might be that humans considered it important that the objects in Figure 6.2 have the same size, whereas the SpCD measures the distribution of colours.

We observe that the SpCD values for Gallery A (Table 6.1) differ greatly from one picture to another. This implies dissimilarity between the pictures, but these differences are not bigger than those for Gallery B (Table 6.2), rather the opposite.

Table 6.9: Ranking of galleries in Figures 6.1–6.4 according to similarity of pictures in gallery

Rank	Gallery	Perceptual value
1	D	3.64
2	C	3.12
3	B	2.40
4	A	1.82

Consider Table 6.10, which gives the results for Figures 6.5–6.7. Humans ranked Gallery F in Figure 6.6 highest, that is, as the gallery with pictures that are most similar to each other. This view correlates with the SpCD measures for this gallery (Table 6.6). Four pictures have the value 0, which means that the SpCD measure found them to be identical to the query picture. The pictures are not identical, but this result shows that both measures found these pictures to be very similar. Humans ranked Gallery G in Figure 6.7 second. This view correlates with the SpCD measures for this gallery (Table 6.7), which are the second highest for the three galleries under consideration. Humans ranked Gallery E in Figure 6.5 third. This view correlates with the SpCD measures for this gallery (Table 6.5), which are the highest for the three galleries under consideration.

Table 6.10: Ranking of galleries in Figures 6.5–6.7 according to similarity of pictures in gallery

Rank	Gallery	Perceptual value
1	F	2.66
2	G	2.24
3	E	2.17

6.2.4 Factors that Determine Similarity

In the last section of the survey, respondents were asked which factor was most important to them when determining the similarity of the pictures in a gallery. Table 6.11 shows the factors that respondents considered important, and the percentage of respondents for each factor.

6.3 Evaluation

Only one gallery, namely Gallery D in Figure 6.4, showed no correlation at all between the SpCD and perceptual similarity in ranking the pictures. This gallery was treated as an outlier as humans failed to rank the picture

Table 6.11: Most important factor when determining similarity of pictures

Rank	**Factor**	**%**
1	Distribution of the objects in the picture	46.46
2	Distribution of the colours in the picture	28.54
3	Objects in the picture	14.39
4	Colours present in the picture	6.31
5	Other: symmetry; both distribution of colours and objects in the picture; subshapes; patterns	4.29

which was used as the query picture correctly. One gallery, namely Gallery F in Figure 6.6, had the same ranking for both the SpCD and perceptual similarity. For four galleries, namely galleries A–C and E (Figures 6.1–6.3 and 6.5), the correlation was high, in that there were more pictures that were ranked the same by both measures than pictures that were not. In the remaining gallery, Gallery G in Figure 6.7, there were more pictures that were ranked differently by both measures than those that were ranked the same.

It is important to evaluate the effectiveness of the SpCD in representing perceptual similarity. Such an evaluation will aid us in determining whether or not the SpCD is consistent with perceptual similarity and direct future research. In this evaluation, we use discounted cumulative gain (DCG) [40], which evaluates the ranking of documents. The key feature in DCG is that highly relevant documents should be ranked higher than the less relevant ones. Since, in this survey, the main focus was on the ranking of pictures, discounted cumulative gain was deemed to be the best method to evaluate the consistency between perceptual similarity and the SpCD. Furthermore, we present the evaluation by the normalized discounted cumulative gain (NDCG) [51], which normalizes the values to lie between 0 and 1, to aid the comparison.

The discounted cumulative gain for a given query is

$$\mathrm{DCG} = \sum_{i=1}^{n} \frac{\mathrm{rating}(i)}{\log_2(1+i)} \ ,$$

where

- n is the number of ranks,
- i is the rank of a picture from 1 (most similar to the query picture) to 5 (least similar), and
- $\mathrm{rating}(i)$ is the value assigned to a picture according to its perceptual similarity, from 5 (most similar) to 1 (least similar).

The ideal discounted cumulative gain (IDCG) for a given query is the DCG according to the perceptual ranking.

The normalization is calculated by dividing the DCG by the IDCG, that is,

$$\text{NDCG} = \frac{\text{DCG}}{\text{IDCG}} \ .$$

For example, Table 6.12 gives the DCG calculation for Table 6.1.

Table 6.12: DCG calculation for Table 6.1

	SpCD (DCG)			Perceptual (IDCG)		
i	**Picture**	**rating(i)**	**DCG**	**Picture**	**rating(i)**	**IDCG**
1	c	5	$\frac{5}{\log_2(1+1)}$	c	5	$\frac{5}{\log_2(1+1)}$
2	d	3	$\frac{3}{\log_2(1+2)}$	b	4	$\frac{4}{\log_2(1+2)}$
3	b	4	$\frac{4}{\log_2(1+3)}$	d	3	$\frac{3}{\log_2(1+3)}$
4	e	2	$\frac{2}{\log_2(1+4)}$	e	2	$\frac{2}{\log_2(1+4)}$
5	a	1	$\frac{1}{\log_2(1+5)}$	a	1	$\frac{1}{\log_2(1+5)}$
	$\text{DCG} = \sum_{i=1}^{5} \frac{\text{rating}(i)}{\log_2(i+1)} = 10.138$			$\text{IDCG} = \sum_{i=1}^{5} \frac{\text{rating}(i)}{\log_2(i+1)} = 10.269$		

Table 6.13 presents the DCG and NDCG values for Tables 6.1 to 6.7. The average NDCG is 0.961. The closer the NDCG value is to 1, the higher the correlation between the ranking of the pictures by the SpCD and by human perception.

Table 6.13: NDCG results

Table	**DCG**	**IDCG**	**NDCG**
6.1	10.138	10.269	0.987
6.2	10.138	10.269	0.987
6.3	10.095	10.269	0.983
6.4	8.388	10.269	0.807
6.5	10.200	10.269	0.993
6.6	10.269	10.269	1.000
6.7	10.006	10.269	0.974
Average NDCG			0.961

6.4 Conclusion

In this chapter, we presented the results of an online survey that we conducted to determine how humans determine the similarity of syntactically generated pictures. We applied the spatial colour distribution descriptor to the same images and we present results which compare whether the human view of similarity is consistent with the selected mathematical similarity measure.

The humans seemed to have very different opinions regarding the similarity of pictures. A reason may be that different people compare pictures using different measures, some placing more emphasis on colour while others place more emphasis on objects. However, the majority of respondents agreed on the similarity of individual pictures compared to the query picture. Most respondents found the given galleries of pictures contained similar pictures, which is very important as this research is about the generation of similar pictures. When comparing the results of the survey with the results of the spatial colour distribution descriptor similarity measure, perceptual similarity seemed to correlate to the spatial colour distribution descriptor measure. This implies that the spatial colour distribution descriptor can be used to judge the similarity of pictures generated by bag context picture grammars and random context picture grammars.

Chapter 7

Conclusion

The need to generate similar pictures syntactically is gaining much interest as a result of applications such as visual password systems that require the use of similar pictures. The problem with many picture grammars available is that they generate an infinite number of pictures without considering the similarity of the pictures. This leads to the creation of many pictures with some that are dissimilar to the selected password picture. Hence much time and resources are wasted in finding the pictures that are similar. This research in the syntactic generation of similar pictures was motivated by the desire to generate in a controlled way a finite number of similar pictures to overcome these problems. We can avoid creating an infinite number of pictures with some different altogether by creating only structurally similar pictures to the query picture. This saves time and ensures the availability of similar pictures for use in visual password systems.

This research has made the following contributions:

- Defined bag context picture grammars and shown some examples of grammars and sets of pictures generated by them.
- Showed how any RCPG can be rewritten as a BCPG, defined a lemma for it and provided a lemma for the cases of RPCPGs, RFCPGs, and CFPGs.
- Developed two syntactic picture generators; BACOPA for processing BCPGs and RACOPA for processing RCPGs.
- Suggested methods for generating structurally similar pictures by using BCPGs and RCPGs. The presented strategies are limiting the number of refinements of the generated pictures, limiting the refinement of some sub-pictures in the pictures and limiting both refinement levels and refinement of sub-pictures. We have also compared the ease of using BCPGs and RCPGs in generating similar pictures.

- We showed that all the methods presented above improve similarity of the generated pictures.

- We investigated how humans perceived the similarity of syntactically generated images and showed that the human view of similarity is consistent with SpCD (the chosen mathematical similarity measure).

Using BCPGs and RCPGs, the presented methods generated in a controlled manner a finite number of similar pictures. The regulated rewriting in both BCPGs and RCPGs has made it possible to control the application of rules to produce only structurally similar pictures. All the presented methods for the generation of similar pictures have shown to improve the similarity of the produced pictures. These methods can be used to generate similar pictures for use in visual password systems or other applications.

Bag context picture grammars defined in this research can be used in the generation of pictures and further research on them. SpCD can be used to measure the similarity of syntactically generated pictures, particularly BCPGs and RCPGs.

This research may lead to several interesting further research options as follows:

- Other picture grammars can adopt the presented methods to generate similar pictures.

- We have only provided a few examples of grammars and picture sets due to time constraints, but other researchers may explore with many more grammars and pictures to see the effectiveness of the methods.

- Researchers can use similar pictures produced by random context picture grammars and bag context picture grammars in visual password systems and investigate their effectiveness.

- Another line of enquiry concerns the bag positions. When one rewrites an RCPG as a BCPG, using the above mentioned lemma, the resultant BCPG has a bag position for each variable of the RCPG. By inspecting the BCPG, one can often reduce the number of bag positions. The question is whether there exist general rules for reducing bag positions.

- It will be interesting to examine if BCPGs are more powerful than RCPGs. For tree grammars, BCTGs are more powerful than RCTGs, that is, there are BCTGs such that no equivalent RCTG does exist. As the formalism of BCPGs has evolved from the RCPG formalism in the same way as BCTG has evolved from RCTG, one naturally asks whether BCPGs are also more powerful than RCPGs.

References

[1] Bhika, C., Ewert, S., Schwartz, R., and Waruhiu, M. (2007). Table-driven context-free picture grammars. *International Journal of Foundations of Computer Science*, 18(6):1151–1160.

[2] Biddle, R., Chiasson, S., and van Oorschot, P. C. (2012). Graphical passwords: Learning from the first twelve years. *ACM Computing Surveys*, 44(4):19:1–19:41.

[3] Bille, P. (2005). A survey on tree edit distance and related problems. *Theoretical Computer Science*, 337(1):217–239.

[4] Boden, N., Atcheson, B., and Ewert, S. (2004). GRIFFIN: An automated GRCPG and IFS processing tool. *South African Computer Journal*, (33):24–37.

[5] Chatzichristofis, S. A., Boutalis, Y. S., and Lux, M. (2009). Img(rummager): An interactive content based image retrieval system. In *Proceedings of the 2009 Second International Workshop on Similarity Search and Applications*, SISAP '09, pages 151–153, Washington, DC, USA. IEEE Computer Society.

[6] Chatzichristofis, S. A., Boutalis, Y. S., and Lux, M. (2010). SpCD — spatial color distribution descriptor — A fuzzy rule based compact composite descriptor appropriate for hand drawn color sketches retrieval. In *ICAART 2010 - Proceedings of the International Conference on Agents and Artificial Intelligence*, volume 1 - Artificial Intelligence, pages 58–63.

[7] Chen, S. and Zhang, K. (2007). An improved algorithm for tree edit distance incorporating structural linearity. In *Computing and Combinatorics*, pages 482–492. Springer.

[8] Culik II, K. and Dube, S. (1993a). Affine automata and related techniques for generation of complex images. *Theoretical Computer Science*, 116(2):373–398.

[9] Culik II, K. and Dube, S. (1993b). L-systems and mutually recursive function systems. *Acta Informatica*, 30(3):279–302.

[10] Dassow, J. and Paun, G. (1989). *Regulated rewriting in formal language theory*. Springer Publishing Company, Incorporated.

[11] Davis, D., Monrose, F., and Reiter, M. K. (2004). On user choice in graphical password schemes. In *Proceedings of the 13th USENIX Security Symposium, August 9-13, 2004, San Diego, CA, USA*, pages 151–164.

[12] Demaine, E. D., Mozes, S., Rossman, B., and Weimann, O. (2009). An optimal decomposition algorithm for tree edit distance. *ACM Transactions on Algorithms (TALG)*, 6(1):2.

[13] Dhamija, R. and Perrig, A. (2000). Déjà Vu: A user study using images for authentication. In *Proceedings of the 9th Conference on USENIX Security Symposium - Volume 9*, SSYM'00, pages 4–4, Berkeley, CA, USA. USENIX Association.

[14] Drewes, F. (2000). Tree-based picture generation. *Theoretical Computer Science*, 246(1):1–51.

[15] Drewes, F. (2001). Tree-based generation of languages of fractals. *Theoretical Computer Science*, 262(1):377–414.

[16] Drewes, F. (2006). *Grammatical Picture Generation - A Tree-Based Approach*. Texts in Theoretical Computer Science. An EATCS Series. Springer.

[17] Drewes, F., du Toit, C., Ewert, S., Högberg, J., van der Merwe, B., and van der Walt, A. P. J. (2005). Random context tree grammars and tree transducers. *South African Computer Journal*, 34:11–25.

[18] Drewes, F., du Toit, C., Ewert, S., van der Merwe, B., and van der Walt, A. (2008). Bag context tree grammars. *Fundamenta Informaticae*, (86):459–480.

[19] Drewes, F., Habel, A., Kreowski, H., and Taubenberger, S. (1995). Generating self-affine fractals by collage grammars. *Theoretical Computer Science*, 145(1&2):159–187.

[20] Drewes, F. and Klempien-Hinrichs, R. (2000). TREEBAG. In *Implementation and Application of Automata, 5th International Conference, CIAA 2000, London, Ontario, Canada, July 24-25, 2000, Revised Papers*, pages 329–330.

[21] Dulucq, S. and Touzet, H. (2005). Decomposition algorithms for the tree edit distance problem. *Journal of Discrete Algorithms*, 3(24):448–471. Combinatorial Pattern Matching (CPM) Special IssueThe 14th annual Symposium on combinatorial Pattern Matching.

[22] Dunphy, P. and Yan, J. (2007). Do background images improve "draw a secret" graphical passwords? In *Proceedings of the 2007 ACM Conference on Computer and Communications Security, CCS 2007, Alexandria, Virginia, USA, October 28-31, 2007*, pages 36–47.

[23] Ewert, S. (1999). *Random Context Picture Grammars*. PhD thesis, University of Stellenbosch.

[24] Ewert, S. (2009). Random context picture grammars: The state of the art. In Drewes, F., Habel, A., Hoffmann, B., and Plump, D., editors, *Manipulation of Graphs, Algebras and Pictures*, pages 135–147. Hohnholt, Bremen.

[25] Ewert, S., Jingili, N., Mpota, L., and Sanders, I. (2019). Bag context picture grammars. *Journal of Computer Languages*, 51:214–221.

[26] Ewert, S. and Rabkin, M. (2012). Generalized random context picture grammars: The state of the art. In *Languages Alive - Essays Dedicated to Jürgen Dassow on the Occasion of His 65th Birthday*, pages 56–74.

[27] Ewert, S. and van der Walt, A. (1998). Generating pictures using random forbidding context. *International Journal of Pattern Recognition and Artificial Intelligence*, 12(7):939–950.

[28] Ewert, S. and van der Walt, A. (1999a). Generating pictures using random permitting context. *International Journal of Pattern Recognition and Artificial Intelligence*, 13(3):339–355.

[29] Ewert, S. and van der Walt, A. (1999b). A hierarchy result for random forbidding context picture grammars. *International Journal of Pattern Recognition and Artificial Intelligence*, 13(7):997–1007.

[30] Ewert, S. and van der Walt, A. P. (1999c). Shrink indecomposable fractals. *Journal of Universal Computer Science*, 5(9):521–531.

[31] Fernau, H., Freund, R., Siromoney, R., and Subramanian, K. G. (2015). Non-isometric contextual array grammars with regular control and local selectors. In *Machines, Computations, and Universality - 7th International Conference, MCU 2015, Famagusta, North Cyprus, September 9-11, 2015, Proceedings*, pages 61–78.

[32] Fernau, H., Paramasivan, M., Schmid, M. L., and Thomas, D. G. (2018). Simple picture processing based on finite automata and regular grammars. *Journal of Computer and System Sciences*, 95:232–258.

[33] Goldberg, J., Hagman, J., and Sazawal, V. (2002). Doodling our way to better authentication. In *Extended Abstracts of the 2002 Conference on Human Factors in Computing Systems, CHI 2002, Minneapolis, Minnesota, USA, April 20-25, 2002*, pages 868–869.

[34] Goldberger, J., Gordon, S., and Greenspan, H. (2003). An efficient image similarity measure based on approximations of KL-divergence between two gaussian mixtures. In *9th IEEE International Conference on Computer Vision (ICCV 2003), 14-17 October 2003, Nice, France*, pages 487–493. IEEE Computer Society.

[35] Hinz, F. (1986). Regular chain code picture languages of nonlinear descriptional complexity. *Lecture Notes in Computer Science*, 223:414–421.

[36] Hinz, F. (1989). Classes of picture languages that cannot be distinguished in the chain code concept and deletion of redundant retreats. *Lecture Notes in Computer Science*, 349:132–143.

[37] Huang, J., Kumar, S., Mitra, M., Zhu, W.-J., and Zabih, R. (1997a). Image indexing using color correlograms. In *Proceedings of the 1997 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 762–768. IEEE Computer Society.

[38] Huang, J., Kumar, S. R., and Mitra, M. (1997b). Combining supervised learning with color correlograms for content-based image retrieval. In *Proceedings of the Fifth ACM International Conference on Multimedia*, pages 325–334. ACM.

[39] Idahosa, J. (2015). *Ogden's Lemma for Random Permitting and Forbidding Context Picture Languages and Table-Driven Context Free Picture Languages*. Master of Science Dissertation, School of Computer Science, University of the Witwatersrand, Johannesburg, South Africa.

[40] Järvelin, K. and Kekäläinen, J. (2000). Ir evaluation methods for retrieving highly relevant documents. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '00, pages 41–48, New York, NY, USA. ACM.

[41] Jermyn, I., Mayer, A. J., Monrose, F., Reiter, M. K., and Rubin, A. D. (1999). The design and analysis of graphical passwords. In *Proceedings of the 8th USENIX Security Symposium, Washington, DC, USA, August 23-26, 1999*.

[42] Jesus, D., Coelho, A., and de Sousa, A. A. (2016). Layered shape grammars for procedural modelling of buildings. *The Visual Computer*, 32(6-8):933–943.

[43] JetBrains (2016). PyCharm: Python ide for professional developers. [online] Available at: https://www.jetbrains.com/pycharm/ [Accessed 15 February 2016].

[44] Jingili, N., Ewert, S., and Sanders, I. (2018a). Syntactic generation of similar pictures. Submitted.

[45] Jingili, N., Ewert, S., and Sanders, I. D. (2018b). Measuring perceptual similarity of syntactically generated pictures. In Rango, F. D., Ören, T. I., and Obaidat, M. S., editors, *Proceedings of 8th International Conference on Simulation and Modeling Methodologies, Technologies and Applications, SIMULTECH 2018, Porto, Portugal, July 29-31, 2018.*, pages 244–255. SciTePress.

[46] Jowers, I., Earl, C., and Stiny, G. (2019). Shapes, structures and shape grammar implementation. *Computer-Aided Design*, 111:80 – 92.

[47] Kiranyaz, S., Birinci, M., and Gabbouj, M. (2010). Perceptual color descriptor based on spatial distribution: A top-down approach. *Image and Vision Computing*, 28(8):1309–1326.

[48] Klein, P. N. (1998). Computing the edit-distance between unrooted ordered trees. In Bilardi, G., Italiano, G. F., Pietracaprina, A., and Pucci, G., editors, *Algorithms - ESA '98, 6th Annual European Symposium, Venice, Italy, August 24-26, 1998, Proceedings*, volume 1461 of *Lecture Notes in Computer Science*, pages 91–102. Springer.

[49] Klempien-Hinrichs, R., Kreowski, H., and Taubenberger, S. (1999). Correct translation of mutually recursive function systems into TOL collage grammars. In *Fundamentals of Computation Theory, 12th International Symposium, FCT '99, Iasi, Romania, August 30 - September 3, 1999, Proceedings*, pages 350–361.

[50] Kruger, H. and Ewert, S. (2006). Translating mutually recursive function systems into generalised random context picture grammars. *South African Computer Journal*, (36):99–109.

[51] Le, Q. V. and Smola, A. J. (2007). Direct optimization of ranking measures. *Computing Research Repository*, abs/0704.3359.

[52] Lee, H. Y., Lee, H. K., and Ha, Y. (2003). Spatial color descriptor for image retrieval and video segmentation. *IEEE Transaction on Multimedia*, 5(3):358–367.

[53] Li, B., Chang, E., and Wu, Y. (2003). Discovery of a perceptual distance function for measuring image similarity. *Multimedia Systems*, 8(6):512–522.

[54] Lindenmayer, A. (1968). Mathematical models for cellular interactions in development I. Filaments with one-sided inputs. *Journal of Theoretical Biology*, 18(3):280–299.

[55] Ma, W.-Y. and Zhang, H. (1998). Benchmarking of image features for content-based retrieval. In *Signals, Systems and Computers, 1998. Conference Record of the Thirty-Second Asilomar Conference on*, volume 1, pages 253–257. IEEE.

[56] Manjunath, B. S. and Ma, W. (1996). Texture features for browsing and retrieval of image data. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 18(8):837–842.

[57] McCormack, J. (1993). Interactive evolution of L-system grammars for computer graphics modelling. *Complex Systems: from Biology to Computation*, pages 118–130.

[58] Mihajlov, M. and Jerman-Blazic, B. (2017). Eye tracking graphical passwords. In *International Conference on Applied Human Factors and Ergonomics*, pages 37–44. Springer.

[59] Murthy, G. V., C., P. K., and Kumar, V. V. (2014). A new model of array grammar for generating connected patterns on an image neighborhood. *CoRR*, abs/1407.8337.

[60] Neumann, D. and Gegenfurtner, K. R. (2006). Image retrieval and perceptual similarity. *ACM Transactions on Applied Perception (TAP)*, 3(1):31–47.

[61] Nivat, M., Saoudi, A., Subramanian, K. G., Siromoney, R., and Dare, V. R. (1991). Puzzle grammars and context-free array grammars. *International Journal of Pattern Recognition and Artificial Intelligence*, 5(5):663–676.

[62] Okundaye, B. (2015). *A Tree Grammar-based Visual Password Scheme*. PhD thesis, School of Computer Science, University of the Witwatersrand, Johannesburg, South Africa.

[63] Okundaye, B., Ewert, S., and Sanders, I. (2013). Determining image similarity from pattern matching of abstract syntax trees of tree picture grammars. In *Proceedings of the Twenty-Fourth Annual Symposium of the Pattern Recognition Association of South Africa*, pages 83–90. PRASA, RobMech, AfLaT.

[64] Okundaye, B., Ewert, S., and Sanders, I. (2014a). A novel approach to visual password schemes using tree picture grammars. In Puttkammer, M.

and Eiselen, R., editors, *Proceedings of the 2014 PRASA, RobMech and AfLaT International Joint Symposium*, pages 247–252. PRASA, RobMech, AfLaT.

[65] Okundaye, B., Ewert, S., and Sanders, I. (2014b). Perceptual similarity of images generated using tree grammars. In *Proceedings of the Annual Conference of the South African Institute for Computer Scientists and Information Technologists (SAICSIT 2014)*, pages 286–296. ACM.

[66] Pawlik, M. and Augsten, N. (2011). RTED: a robust algorithm for the tree edit distance. *Proceedings of the VLDB Endowment*, 5(4):334–345.

[67] Pawlik, M. and Augsten, N. (2015). Efficient computation of the tree edit distance. *ACM Transactions on Database Systems (TODS)*, 40(1):3.

[68] Pawlik, M. and Augsten, N. (2016). Tree edit distance: Robust and memory-efficient. *Inf. Syst.*, 56:157–173.

[69] Prusinkiewicz, P. (1986a). Applications of l-systems to computer imagery. In *Graph-Grammars and Their Application to Computer Science, 3rd International Workshop, Warrenton, Virginia, USA, December 2-6, 1986*, pages 534–548.

[70] Prusinkiewicz, P. (1986b). Graphical applications of L-systems. In *Proceedings of Graphics Interface*, volume 86, pages 247–253.

[71] Prusinkiewicz, P. (1999). A look at the visual modeling of plants using L-systems. *Agronomie*, 19(3/4):211–224.

[72] Prusinkiewicz, P., Hanan, J., and Měch, R. (2000). An L-system-based plant modeling language. In *Applications of Graph Transformations with Industrial Relevance*, pages 395–410. Springer.

[73] Prusinkiewicz, P. and Lindenmayer, A. (1990). *Graphical modeling using L-systems*. Springer.

[74] Real-User-Corporation (2012). The science behind passfaces. [online] Available at: http://www.realuser.com/published/The Science Behind Passfaces.pdf.[Accessed 15 January 2018].

[75] Room, P., Hanan, J., and Prusinkiewicz, P. (1996). Virtual plants: new perspectives for ecologists, pathologists and agricultural scientists. *Trends in Plant Science*, 1(1):33–38.

[76] Ruiz-Montiel, M., Pérez-de-la-Cruz, J., Mandow, L., and López-Romero, F. (2016). Fuzzy shape grammars. *Progress in Artificial Intelligence*, 5(1):47–53.

[77] Siromoney, G., Siromoney, R., and Krithivasan, K. (1973). Picture languages with array rewriting rules. *Information and Control*, 22(5):447–470.

[78] Siromoney, G., Siromoney, R., and Krithivasan, K. (1974). Array grammars and kolam. *Computer Graphics and Image Processing*, 3(1):63–82.

[79] Siromoney, R., Huq, A., Chandrasekaran, M., and Subramanian, K. G. (1992). Stochastic puzzle grammars. *International Journal of Pattern Recognition and Artificial Intelligence*, 6(2&3):257–273.

[80] Stiny, G. and Gips, J. (1971). Shape grammars and the generative specification of painting and sculpture. In *IFIP Congress (2)*, pages 1460–1465.

[81] Subramanian, K. G., Isawasan, P., Venkat, I., Pan, L., and Nagar, A. (2014). Array P systems with permitting features. *Journal of Computational Science*, 5(2):243–250.

[82] Subramanian, K. G., Saravanan, R., and Chandra, P. H. (2006). Cooperating basic puzzle grammar systems. In *Combinatorial Image Analysis, 11th International Workshop, IWCIA 2006, Berlin, Germany, June 19-21, 2006, Proceedings*, pages 354–360.

[83] Subramanian, K. G., Siromoney, R., and Dare, V. R. (1995). Basic puzzle languages. *International Journal of Pattern Recognition and Artificial Intelligence*, 9(5):763–775.

[84] Subramanian, K. G., Siromoney, R., Dare, V. R., and Saoudi, A. (1992). Basic puzzle grammars and isosceles right triangles. *International Journal of Pattern Recognition and Artificial Intelligence*, 6(5):799–816.

[85] Subramanian, K. G., Sriram, S., Song, B., and Pan, L. (2018). An overview of 2d picture array generating models based on membrane computing. In *Reversibility and Universality, Essays Presented to Kenichi Morita on the Occasion of his 70th Birthday.*, pages 333–356.

[86] Subramanian, K. G., Venkat, I., and Csuhaj-Varjú, E. (2013). On the power of permitting features in cooperating context-free array grammar systems. *Discrete Applied Mathematics*, 161(15):2328–2335.

[87] Sun, J., Zhang, X., Cui, J., and Zhou, L. (2006). Image retrieval based on color distribution entropy. *Pattern Recognition Letters*, 27(10):1122–1126.

[88] Swain, M. J. and Ballard, D. H. (1991). Color indexing. *International Journal of Computer Vision*, 7(1):11–32.

[89] Tai, K.-C. (1979). The tree-to-tree correction problem. *Journal of the ACM (JACM)*, 26(3):422–433.

[90] Tamura, H., Mori, S., and Yamawaki, T. (1978). Textural features corresponding to visual perception. *IEEE Transaction on Systems, Man, and Cybernetics*, 8(6):460–473.

[91] Tkachova, D. (2013). *Necessary Conditions for Random Context Galleries*. Bachelor of Science with Honours Research Report, School of Computer Science, University of the Witwatersrand, Johannesburg, South Africa.

[92] van der Walt, A. and Ewert, S. (2003). A property of random context picture grammars. *Theoretical Computer Science*, 301(1):313–320.

[93] van Oorschot, P. C. and Thorpe, J. (2011). Exploiting predictability in click-based graphical passwords. *Journal of Computer Security*, 19(4):669–702.

[94] Venkat, I., Thamburaj, R., Subramanian, K. G., and Wilde, P. D. (2018). Generation of kolam-designs based on contextual array P systems. In *Diagrammatic Representation and Inference - 10th International Conference, Diagrams 2018, Edinburgh, UK, June 18-22, 2018, Proceedings*, pages 79–86.

[95] Wiedenbeck, S., Waters, J., Sobrado, L., and Birget, J. (2006). Design and evaluation of a shoulder-surfing resistant graphical password scheme. In Celentano, A., editor, *Proceedings of the Working Conference on Advanced Visual Interfaces, AVI 2006, Venezia, Italy, May 23-26, 2006*, pages 177–184. ACM Press.

[96] Worth, P. and Stepney, S. (2005). Growing music: musical interpretations of L-systems. In *Applications of Evolutionary Computing*, pages 545–550. Springer.

[97] Yamamoto, H., Iwasa, H., Yokoya, N., and Takemura, H. (1999). Content-based similarity retrieval of images based on spatial color distributions. In *1oth International Conference on Image Analysis and Processing (ICIAP 1999), 27-29 September 1999, Venice, Italy*, pages 951–956. IEEE Computer Society.

[98] Zhang, K. and Shasha, D. (1989). Simple fast algorithms for the editing distance between trees and related problems. *SIAM Journal on Computing*, 18(6):1245–1262.

[99] Zhou, X. S. and Huang, T. S. (2003). Relevance feedback in image retrieval: A comprehensive review. *Multimedia Systems*, 8(6):536–544.

Appendix A

Ethics Clearance

Research Office

HUMAN RESEARCH ETHICS COMMITTEE (NON-MEDICAL)
R14/49 Jingili

CLEARANCE CERTIFICATE	**PROTOCOL NUMBER: H16/02/10**
PROJECT TITLE	Syntactic generation of similiar pictures
INVESTIGATOR(S)	Mrs N Jingili
SCHOOL/DEPARTMENT	Computer Science and Applied Mathematics/
DATE CONSIDERED	19 February 2016
DECISION OF THE COMMITTEE	Approved unconditionally
EXPIRY DATE	15 March 2019

DATE 16 March 2016 **CHAIRPERSON** _____ (Professor J Knight)

cc: Supervisor : Professor S Ewert

DECLARATION OF INVESTIGATOR(S)

To be completed in duplicate and **ONE COPY** returned to the Secretary at Room 10005, 10th Floor, Senate House, University.

I/We fully understand the conditions under which I am/we are authorized to carry out the abovementioned research and I/we guarantee to ensure compliance with these conditions. Should any departure to be contemplated from the research procedure as approved I/we undertake to resubmit the protocol to the Committee. **I agree to completion of a yearly progress report.**

_____ 23 / 03 / 2016
Signature **Date**

PLEASE QUOTE THE PROTOCOL NUMBER ON ALL ENQUIRIES

Appendix B

Sample Online Survey

Good day,

I, Nuru Jingili, a PhD student from the School of Computer Science and Applied Mathematics, University of the Witwatersrand, Johannesburg, invite you to participate in a survey for the research project entitled Syntactic Generation of Similar Pictures.

As part of my research, I am required to collect information on how people perceive the similarity of generated pictures.

The questionnaire has twelve questions and should take about 10 minutes to complete.

Your participation is entirely voluntary. You don't have to participate, and you are free to terminate your participation at any moment. You also don't have to answer any question you don't want to answer. Your participation is confidential, your name or contact details will not be attached to the questionnaire.

The information provided by you in this questionnaire will be used for research purposes only. It will not be used in a manner that would allow identification of your responses.

Completing and submitting this questionnaire indicates your consent to participate in this survey.

If you have any questions or concerns, please feel free to contact me (see below for contact information).

Thank you,

Nuru Jingili

Contact Details:

Nuru Jingili
Email: nurujingili@yahoo.com or 1363750@students.wits.ac.za

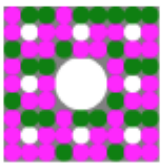
Or you may contact my supervisors,

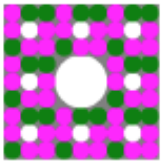
Prof Sigrid Ewert
Email: Sigrid.Ewert@wits.ac.za

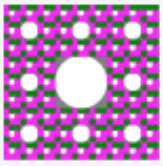
Prof Ian Sanders
Email: sandeid@unisa.ac.za

OK

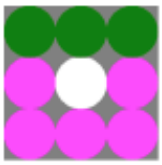
1. Consider the picture shown below. Now rank the pictures that follow in terms of how similar you feel they are to the given picture. In the ranking, 1 should be selected for the most similar picture and 5 for the least similar picture





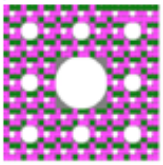


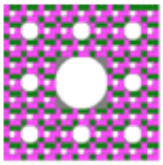
2. Consider the picture shown below. Now rank the pictures that follow in terms of how similar you feel they are to the given picture. In the ranking, 1 should be selected for the most similar picture and 5 for the least similar picture

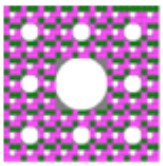


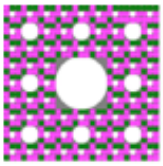
3. Consider the picture shown below. Now rank the pictures that follow in terms of how similar you feel they are to the given picture. In the ranking, 1 should be selected for the most similar picture and 5 for the least similar picture

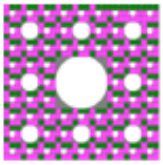
4. Consider the picture shown below. Now rank the pictures that follow in terms of how similar you feel they are to the given picture. In the ranking, 1 should be selected for the most similar picture and 5 for the least similar picture

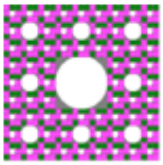






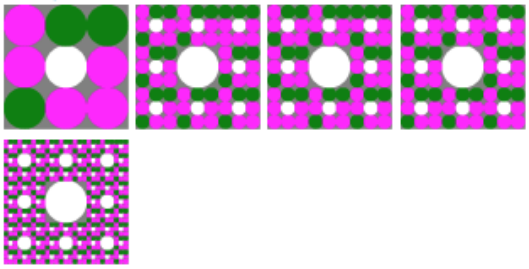




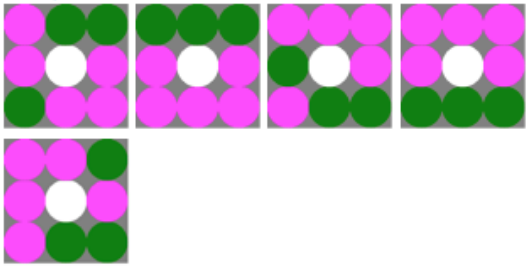


5. For each of the galleries shown below, please select the statement that best describes the similarity of the pictures within that gallery.

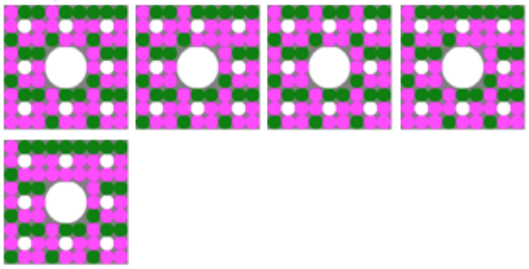
Gallery A



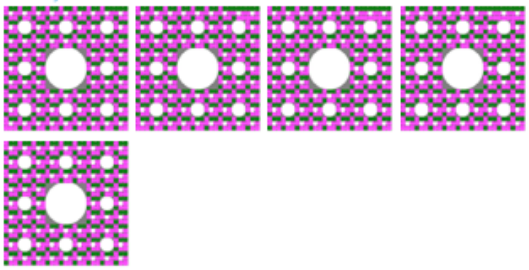
Gallery B



Gallery C



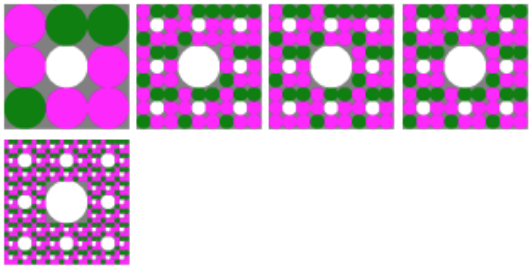
Gallery D



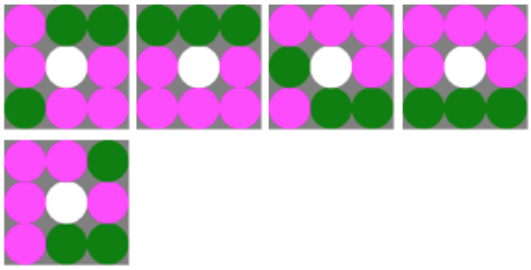
	Not at all similar	Somehow similar	Similar	Very similar	Identical
Gallery A	○	○	○	○	○
Gallery B	○	○	○	○	○
Gallery C	○	○	○	○	○
Gallery D	○	○	○	○	○

6. Rank the galleries shown below from the gallery containing pictures that are more similar to each other to the gallery containing pictures that are least similar to each other.

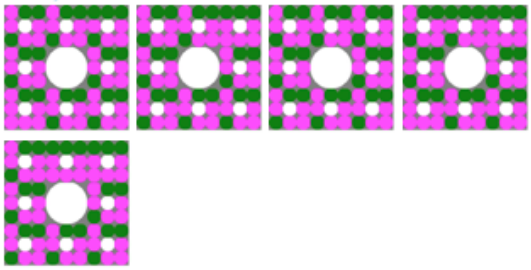
Gallery A



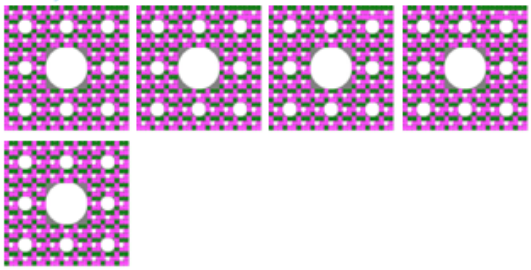
Gallery B



Gallery C



Gallery D



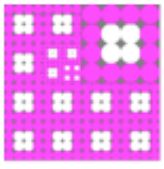
Gallery A

Gallery B

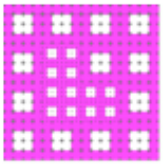
Gallery C

Gallery D

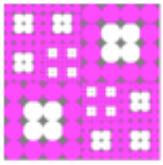
7. Consider the picture shown below. Now rank the pictures that follow in terms of how similar you feel they are to the given picture. In the ranking, 1 should be selected for the most similar picture and 5 for the least similar picture



8. Consider the picture shown below. Now rank the pictures that follow in terms of how similar you feel they are to the given picture. In the ranking, 1 should be selected for the most similar picture and 5 for the least similar picture

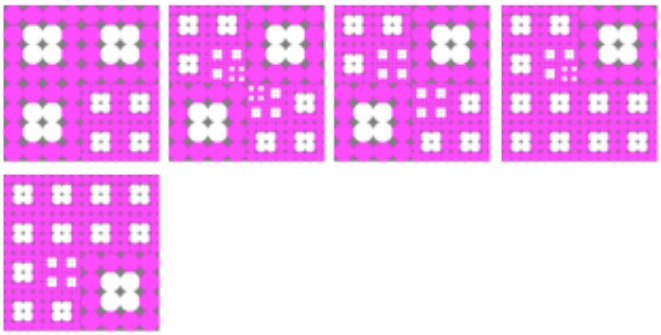


9. Consider the picture shown below. Now rank the pictures that follow in terms of how similar you feel they are to the given picture. In the ranking, 1 should be selected for the most similar picture and 5 for the least similar picture

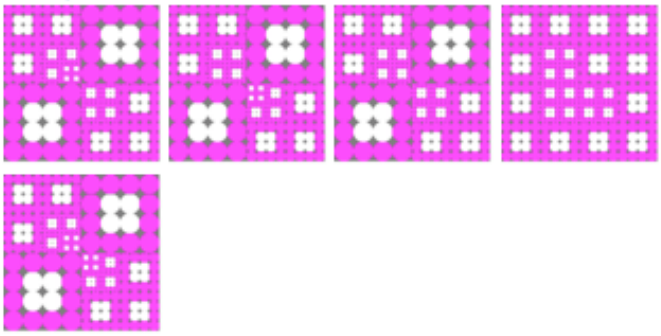


10. For each of the galleries shown below, please select the statement that best describes the similarity of the pictures within that gallery.

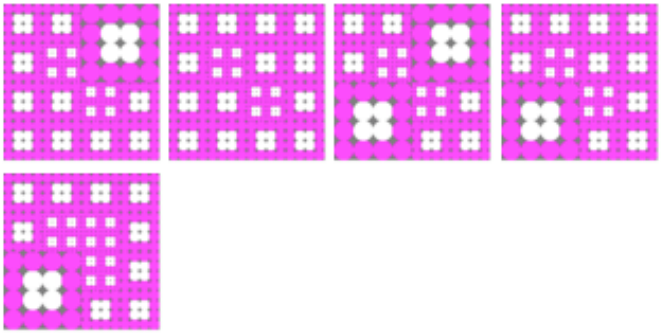
Gallery E



Gallery F



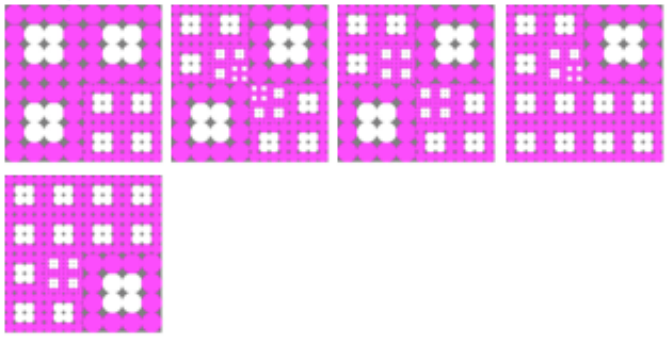
Gallery G



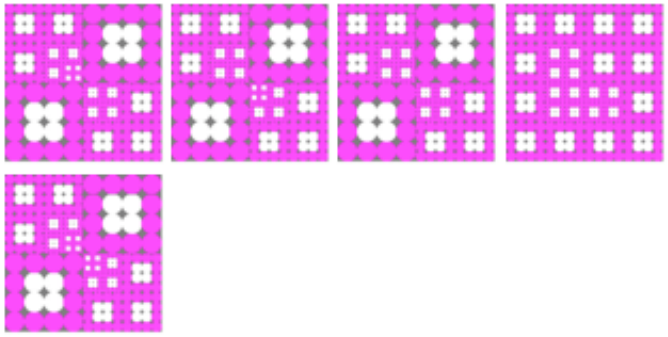
	Not at all similar	Somehow similar	Similar	Very similar	Identical
Gallery E	○	○	○	○	○
Gallery F	○	○	○	○	○
Gallery G	○	○	○	○	○

11. Rank the galleries below from the gallery containing pictures that are more similar to each other to the gallery containing pictures that are least similar to each other.

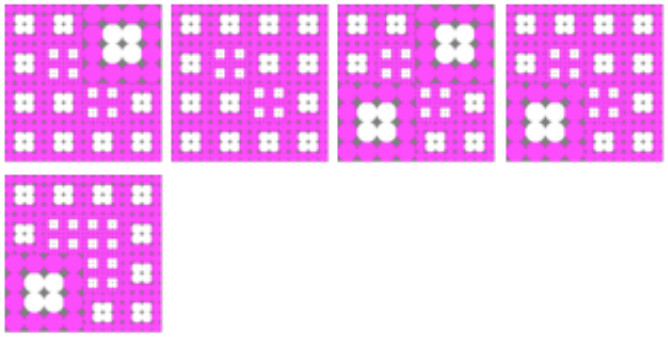
Gallery E



Gallery F



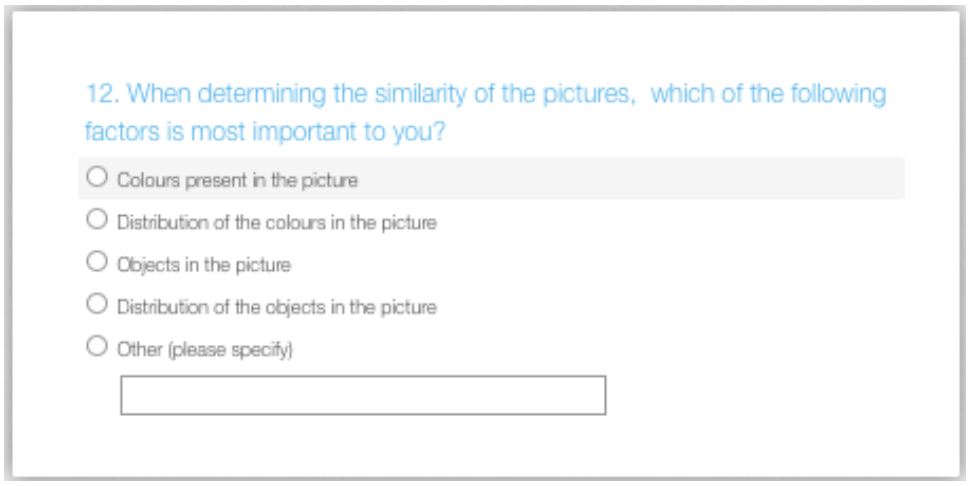
Gallery G

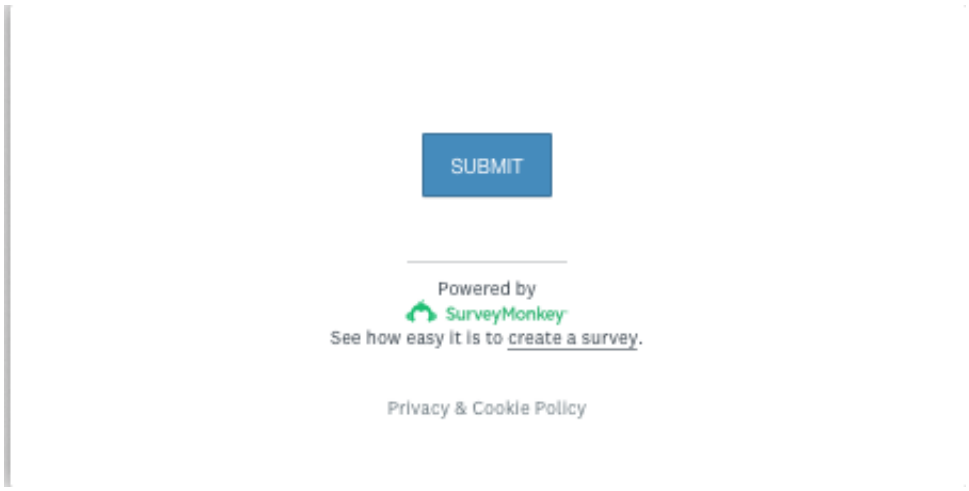


Gallery E

Gallery F

Gallery G


12. When determining the similarity of the pictures, which of the following factors is most important to you?
Colours present in the picture
Distribution of the colours in the picture
Objects in the picture
Distribution of the objects in the picture
Other (please specify)


SUBMIT
Powered by
SurveyMonkey
See how easy it is to create a survey.
Privacy & Cookie Policy

Appendix C

Syntactic Picture Generators

C.1 Introduction

The chapter is a presentation of the applications that use bag context picture grammars and random context picture grammars for processing, generation and visualisation of pictures. Manual processing of grammars in formal languages is very time consuming and prone to errors. It is for this reason that several researchers have developed applications for processing grammars as they offer quick processing of grammar rules and provide quick feedback to the user [4, 20].

In this paper, we present two grammar processing tools namely, BACOPA (bag context picture application) and RACOPA (random context picture application) that process BCPGs and RCPGs respectively. BACOPA and RACOPA were mainly developed to simplify the processing of the grammars, to minimise errors when writing and modifying grammars and to provide quick visual results. Since this research involves the writing and testing of grammars on numerous times, manual processing would have proved to be time-consuming and prone to errors. Thus the use of these applications helps in exploring and demonstrating the usefulness of both BCPGs and RCPGs.

Moreover BACOPA and RACOPA can be used to teach formal grammars, particularly BCPGs and RCPGs. This in turn will go a long way in helping students view the processing of grammars, modify grammars as necessary to obtain desired results and get quick feedback. Furthermore, the applications can act as a framework for the creation of other applications for picture generation.

The rest of the paper is structured as follows: In Section 2, we give background information on picture grammars, in particular BCPGs, on mathematical similarity and perceptual similarity. Section 3 presents the methods we used to generate structurally similar pictures. Section 4 presents the mathematical similarity of the generated pictures. In Section 5, we

present the results of the online survey used to determine the perceptual similarity and analyse the relationship between the perceptual similarity and mathematical similarity. In Section 7 we provide the conclusion.

C.2 Design Considerations

C.2.1 Assumptions

The user of the BACOPA and RACOPA is aware of basic computer operations to follow simple instructions on running a program.

C.2.2 Software Development Methods

The study involved the use of prototyping in the development of the applications. Prototyping made it easy to start with few functions, test the system, get user feedback and keep on adding more functions until we reached the level of satisfaction regarding the application.

C.2.3 System Environment

The application was implemented using Python. Python is operated under the open source license and runs on all major operating systems. Hence the applications can run on any machine with Python. The applications do not need to be installed.

C.2.4 User Interface

The application has a text-based user interface which was implemented using Python. As part of its functions, the application reads grammars from a file. Meanwhile, the user of the system occasionally needs to rewrite or change the grammar in a file. The text-based user interface is also useful for improving the appearance of terminals, as users can change colours, shapes and textures of terminals by specifying that in the program.

C.2.5 Integrated Development Environment (IDE)

The easiest way of running the applications is to install an integrated development environment (IDE). PyCharm [43] was used in the development and implementation of the applications. This is because it was easy to install and navigate through files, make changes and results can be viewed in the same window. To run the applications, a user selects the code file and clicks a start button or right clicks the file and selects run.

C.2.6 Constraints

This section presents the limitations of BACOPA and RACOPA.

- BACOPA can generate pictures using BCPGs only.
- RACOPA can generate pictures using RCPGs only.
- Both applications do not have a graphical user interface. Although the programs do not require many operations, novice Python users might find the applications difficult to use.
- The systems have been implemented using Python. Although the user may use the application without being a Python expert, knowledge of it is necessary to change some functions of the system, for example, colours, shapes and textures.

C.3 BCPG Picture Generator

C.3.1 System Design

BACOPA consists of two Python files, the first file named `bagcontext.py` and the second file named `bcgrammar.py`. The `bagcontext.py` file contains all the code for the generation of pictures while the `bcgrammar.py` file contains the initial bag and grammar rules that are executed by the program. The application can be initialised by loading the file `bagcontext.py`. Each time you run the program, it creates a .png file of each pictorial form and saves it in the folder where the program is stored. A user can change the folder where they want the pictures to be stored in the `bagcontext.py` file. Running the grammar several times allows the generation of a variety of images for that grammar. The user may change the grammar as they wish to in the `bcgrammar.py` file and run the program again. An example of initial bag and grammar rules in `bcgrammar.py` is shown in Figure C.1.

C.3.2 BACOPA Control Flow

This section shows how BACOPA works. The activity diagram presented in Figure C.2 shows the flow of activities in the program.

C.3.3 System Output

BACOPA provides the following output:

- Creates images of all pictorial forms and saves them in a folder. By default, this is the same folder where the application files reside.

```
from numpy import inf
initialbag=[1,0,0,0]
grammar=[["S",("B","A","A","B"),(1,0,0,0),(inf,inf,0,inf),(-1,2,0,0)],
["A",("F"),(0,1,0,0),(0,inf,0,0),(0,-1,0,1)],
["A",("C"),(0,0,0,0),(0,inf,inf,0),(0,-1,1,0)],
["A",("w"),(0,0,0,1),(0,inf,inf,inf),(0,-1,0,0)],
["C",("S"),(0,0,0,0),(inf,0,inf,inf),(1,0,-1,0)],
["B",("b","b","b","b","w","b","b","b","b"),(0,0,0,0),
(inf,inf,inf,inf),(0,0,0,0)],
["F",("w"),(0,0,0,0),(0,0,inf,inf),(0,0,0,-1)]
]
```

Figure C.1: The initial bag and grammar rules as they appear in `bcgrammar.py` file

- Prints statements that inform a user of what the system is doing (for example see Figure C.3). The printed statements assist the user in tracing back how the system derived the pictures. The information that is printed by the system comprises of;
  - current labels in the pictorial form,
  - the selected label,
  - the chosen rule,
  - status of the bag. The result is True if the bag is within limits and False otherwise.
  - the updated bag values
  - the name of an image created and its saved location.

C.4 RCPG Picture Generator

C.4.1 System Design

As in BACOPA, RACOPA also consists of two files namely `randomcontext.py` which contains the code and `rcgrammar.py` which comprises of the grammar rules. The application is initialised by executing the file `randomcontext.py`. Loading the `randomcontext.py` file creates a .png file of each transformation of the picture and saves it in the folder. Loading the `randomcontext.py` several times allows generation of different pictures by the same grammar. The `rcgrammar.py` file can be rewritten to change the grammar. A sample of grammar rules in `rcgrammar.py` is shown in Figure C.4.

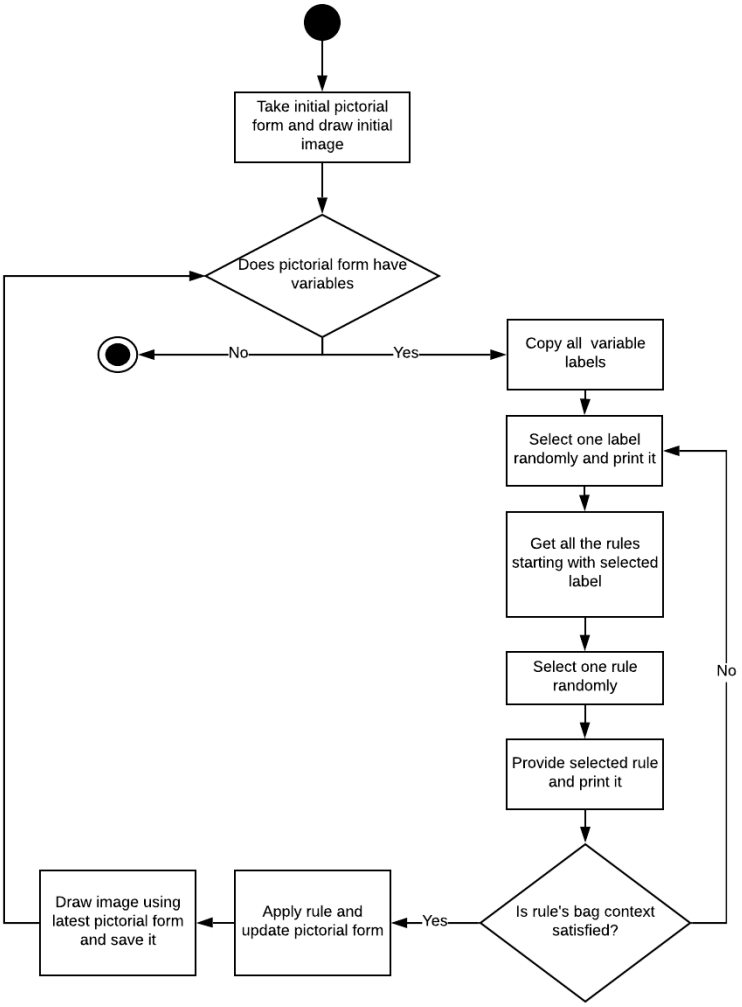


Figure C.2: Activity diagram: bag context picture generator

```
Saved initial file to /Desktop/BagContext/initial.png
Current labels:  ['S']
selected_label:  S
Selected rule:  ['S', ('B', 'A', 'A', 'B'), (1, 0, 0, 0),
(inf, inf, 0, inf), (-1, 2, 0, 0)]
Current Bag:  [1, 0, 0, 0]
results: [True, True, True, True]
Resulting bag: (0, 2, 0, 0)
Saved to  /Desktop/BagContext/image1.png
Current labels:  ['B', 'A', 'A', 'B']
selected_label:  A
Selected rule:  ['A', 'F', (0, 1, 0, 0), (0, inf, 0, 0),
(0, -1, 0, 1)]
Current Bag:  (0, 2, 0, 0)
results: [True, True, True, True]
Resulting bag: (0, 1, 0, 1)
Saved to  /Desktop/BagContext/image2.png
Current labels:  ['B', 'F', 'A', 'B']
selected_label:  A
Selected rule:  ['A', 'w', (0, 0, 0, 1), (0, inf, inf, inf),
(0, -1, 0, 0)]
Current Bag:  (0, 1, 0, 1)
results: [True, True, True, True]
Resulting bag: (0, 0, 0, 1)
Saved to  /Desktop/BagContext/image3.png
```

Figure C.3: An example of the output of BACOPA application

```
grammar=[["S",("B","A","A","B"),((),("C"))],
["A",("F"),((),("S","F","C"))],
["A",("C"),((),("S","F"))],
["A",("w"),(("F"),("S"))],
["C",("S"),((),("A"))],
["B",("b","b","b","b","w","b","b","b","b"),((),())],
["F",("w"),((),("A","S"))],
]
```

Figure C.4: The grammar rules as they appear in `rcgrammar.py` file

C.4.2 RACOPA Control Flow

This section shows how RACOPA works. The flow of activities in the program is shown in the Figure C.5.

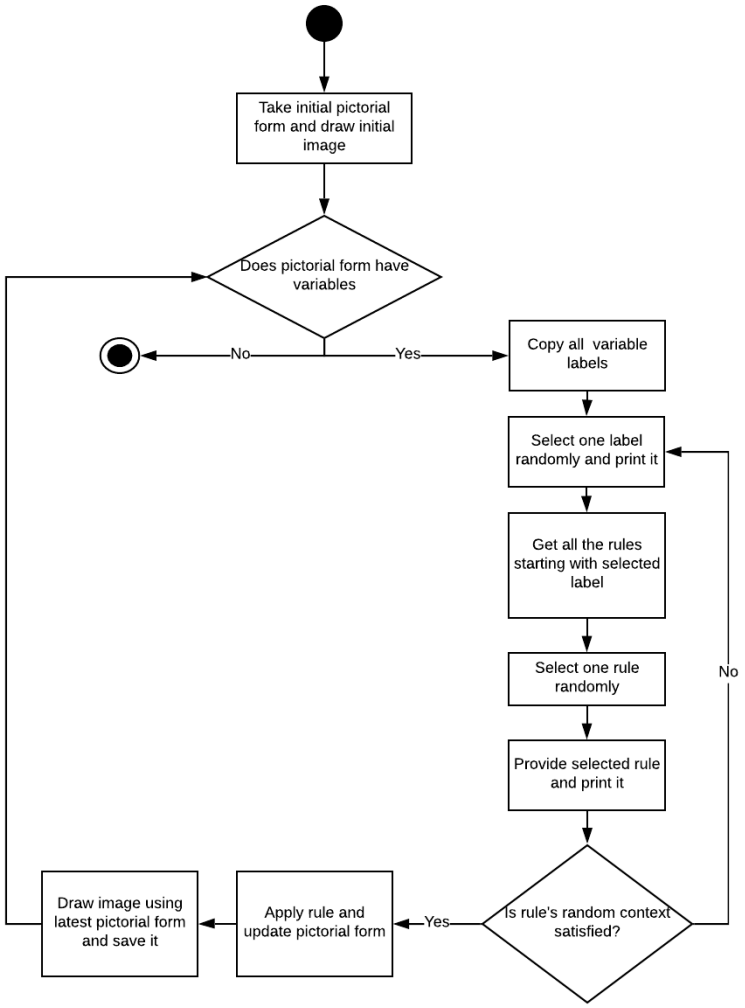


Figure C.5: Activity diagram: random context picture generator

C.4.3 System Output

RACOPA provides the following output:

- Creates pictures of all transformations and saves them in a folder. By default this is the same folder where application files reside.

```
Saved initial file to /Desktop/RandomContext/initial.png
Current labels:  ['S']
selected_label:  S
Selected rule: ['S', ('B', 'A', 'A', 'B'), (), 'C')]
Saved to /Desktop/RandomContext/image1.png
Current labels:  ['B', 'A', 'A', 'B']
selected_label:  A
Selected rule: ['A', 'C', (), ('S', 'F'))]
Saved to  /Desktop/RandomContext/image2.png
Current labels:  ['B', 'C', 'A', 'B']
selected_label:  C
Selected rule: ['C', 'S', (), 'A')]
Saved to /Desktop/RandomContext/image3.png
Current labels:  ['B', 'C', 'A', 'B']
selected_label:  A
Selected rule: ['A', 'F', (), ('S', 'F', 'C'))]
Saved to /Desktop/RandomContext/image4.png
```

Figure C.6: An example of the output of the RACOPA application

- Prints statements that inform the user of what the system is doing. The printed statements provide a footprint of picture derivation. Figure C.6 is a screenshot of the printed statements. The following is the information that is printed by the system;
  - current labels present in the pictorial form,
  - the selected label,
  - the selected rule,
  - the name of the image created and its saved location.

C.5 Pictures Generated by BACOPA and RACOPA

This section presents a sample of pictures produced by both BACOPA and RACOPA. Figure C.7 shows some of the pictures that have been generated by BACOPA and RACOPA.

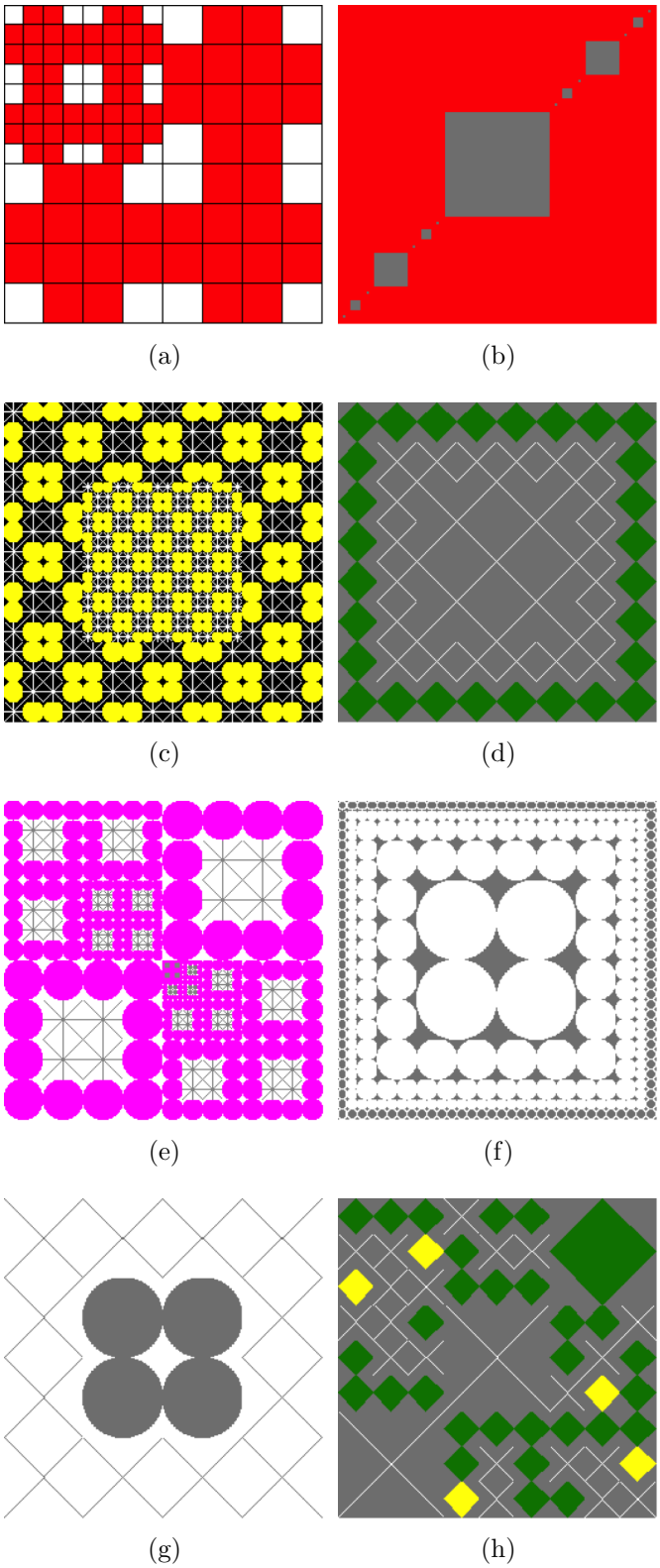


Figure C.7: Some pictures generated by BACOPA and RACOPA

C.6 Future Work

A graphical user interface of the system should be developed to make the system user-friendly. Both applications should be packaged as one application and users should have an option of selecting BCPGs or RCPGs for picture generation.

Appendix D

Visual Password Systems

In visual password systems, pictures are used as passwords. For this research, we generated structurally similar pictures for use in a Passfaces type of visual password system where a user selects a password image among distractor images.

Distractor images are pictures that are similar to a password picture. In this chapter, we present the design of a visual password system that uses the presented methods of generation of similar pictures to generate distractor pictures. The motivation for the given design is the desire to improve the research on the use of grammars for the production of similar images by Okundaye et al. [64]. In the improved design, similar pictures can be generated on a clients computer or on the computer where the visual password system is installed in the case of a networked environment. The visual password system removes the need for creating many pictures then find similar pictures. Moreover, there is no need to have an extensive database of pictures as only the password picture and its distractor images are stored for each user.

D.1 Design of the Visual Password System

The proposed system consists of two subsystems, namely the visual password system and the bag context picture application (BACOPA). Figure D.1 shows the workflow of the proposed visual password system. In the beginning, the system displays a login or register page. Users can provide a username if they are registered. On the other hand, if they are not registered, users are required to register to access the system. To register, the user selects a password image from the gallery of displayed pictures. An example gallery is shown in Figure D.2. Each of the images displayed is associated with a particular grammar. When the user selects an image, the visual password system sends the corresponding grammar to BACOPA. BACOPA records the grammar and generates nine distinct pictures. The pictures are then

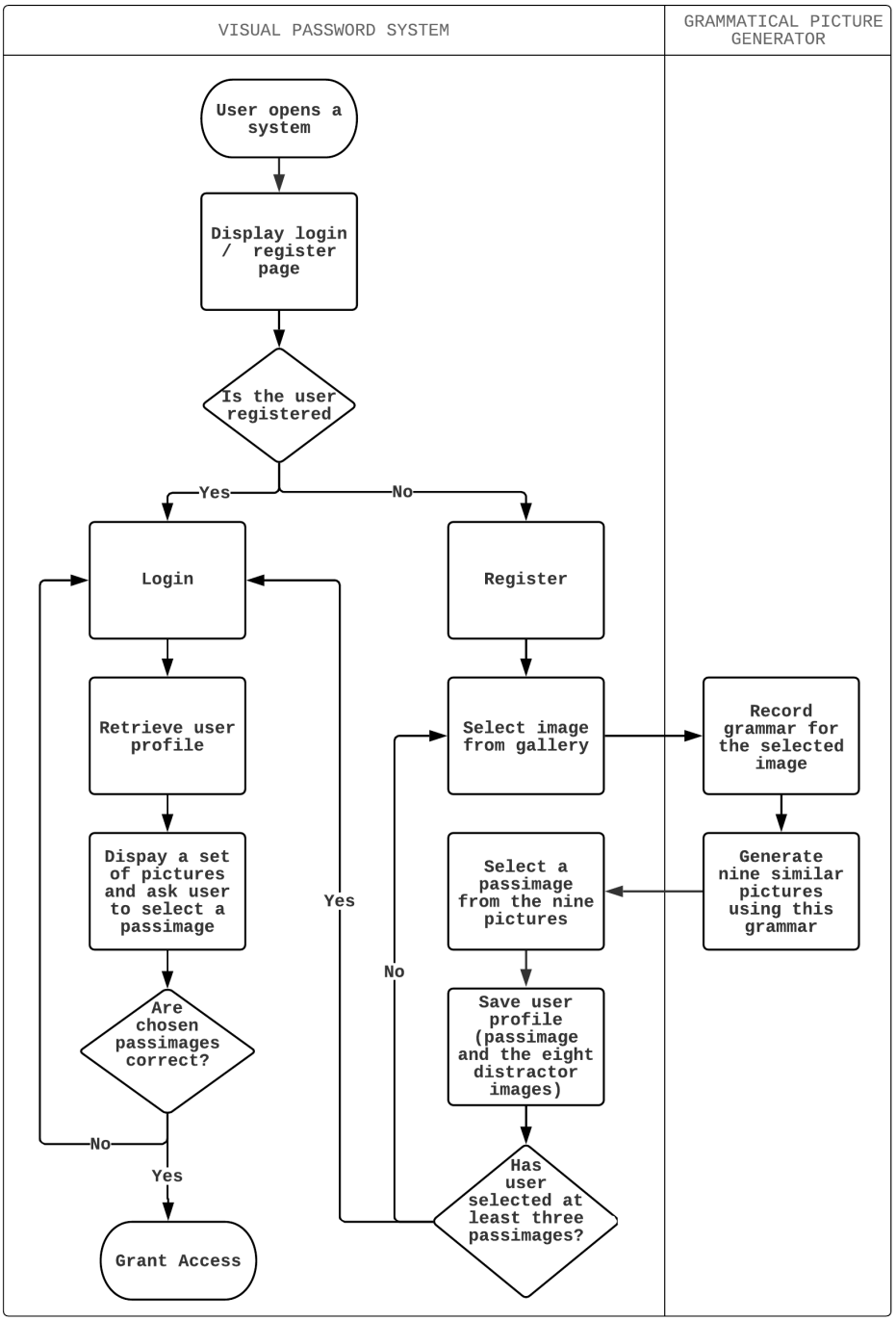


Figure D.1: Visual password system workflow

presented to the user for them to select their password image. Figures D.3, D.4 and D.5 show the example password selection screens. After selection, the image chosen becomes the password, and the remaining eight images become the distractors. Then these pictures are saved in the user profile in the visual password systems. This process is repeated two more times. For each password selection screen, eight distractor images are needed to display them on a 3×3 grid (see Figures D.3, D.4 and D.5). In a basic system, the user needs to have at least three images as password, but for maximum security, the user can select up to twelve password images [64]. The visual password systems should allow the user to specify how many password images they want. The more password images, the more authentication screens, which in turns means higher security as it makes shoulder surfing difficult [64]. After selection of the required password images by the system, the user becomes registered and redirected to the login screen.

To log in to the system, the user provides the username. The visual password system checks if the username exists, if the username exists the visual password system retrieves the user profile and displays a set of pictures on a 3×3 grid screen and asks the user to select their password image (for example, Figures D.3, D.4 and D.5). The password image and the corresponding eight distractor images are randomly placed on the screen. This process is repeated two more times. The visual password system then grants access if all the selected password images were correct. Alternatively, the visual password system denies access if any of the chosen password images was wrong and redirects the user to the login screen. Allowing a user to finish authentication and redirecting them to the login screen leaves the unauthorised user unaware of where their mistake is.

D.2 Conclusion

In this chapter, we presented the design of a Passfaces type of visual password system that uses pictures generated by bag context picture grammars using proposed methods of generation of similar images in Chapter 4. We provided a workflow diagram that shows how the system works and example authentication steps. The given system design offers an improvement on previous research by Okundaye [62] by generating only similar pictures dynamically and thus does not need the use of mathematical similarity measures to find related images. Moreover, the proposed system uses minimal space for storing images as it dynamically generates the required images only. Another merit of the presented design is the lack of bias in selecting images as the system is using abstract pictures. It will be interesting to develop a fully working prototype for the presented design.

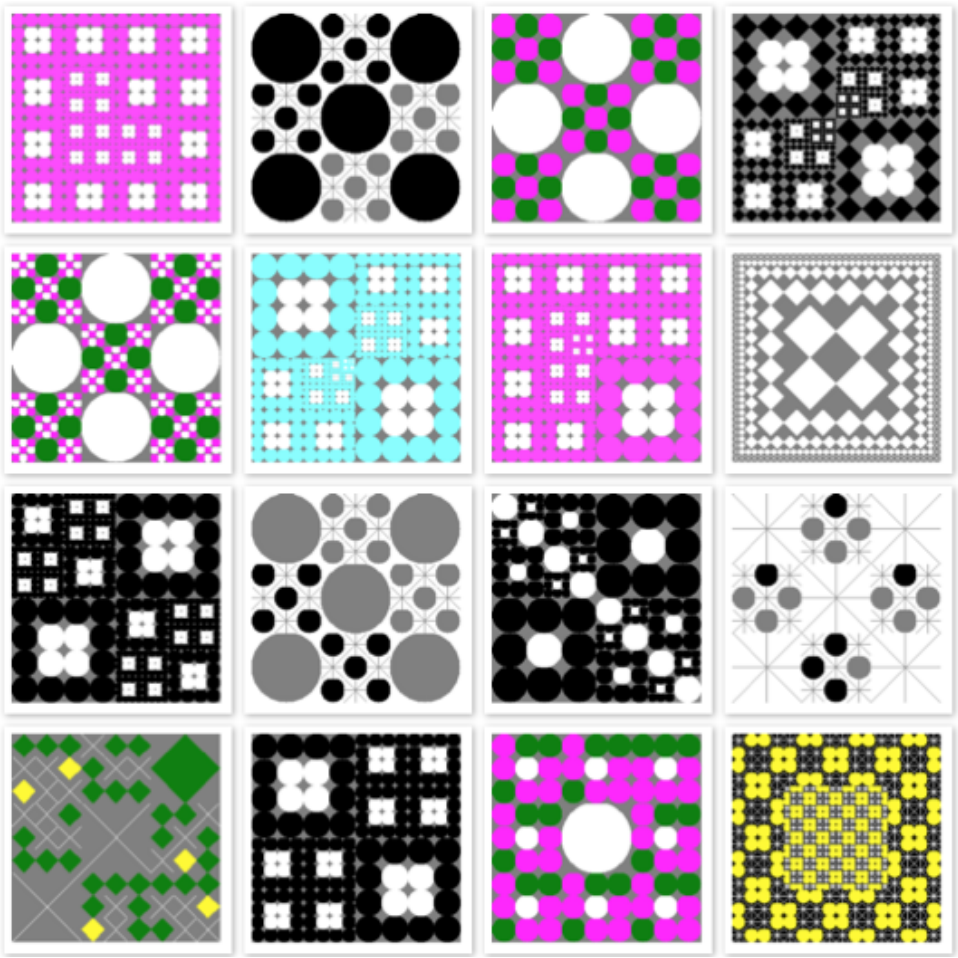


Figure D.2: Example of the first password selection screen

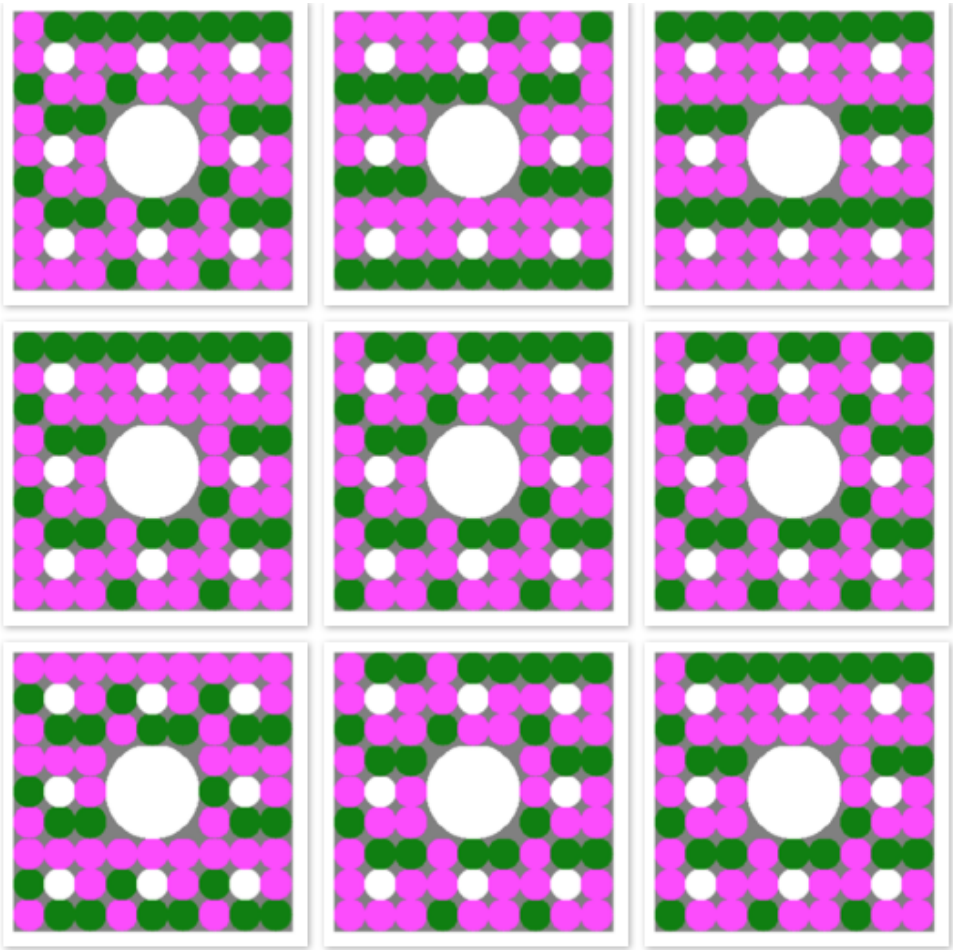


Figure D.3: Example of second password selection screen

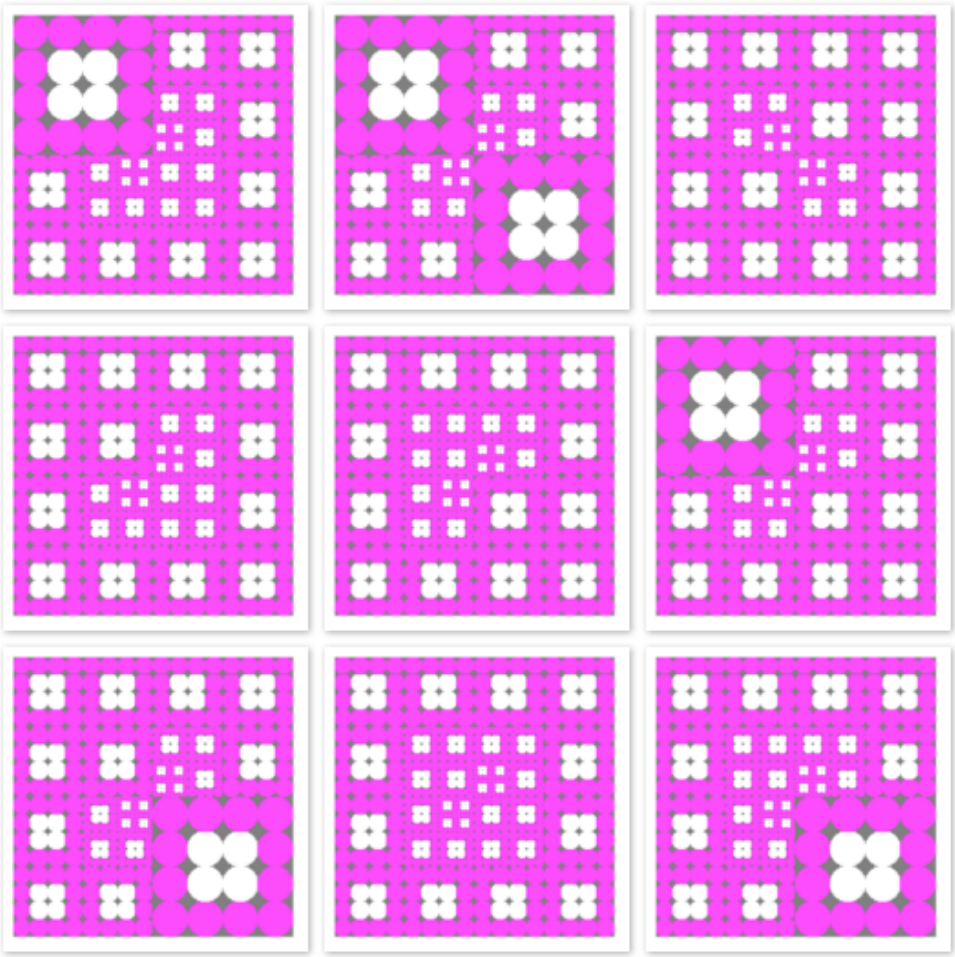


Figure D.4: Example of third password selection screen

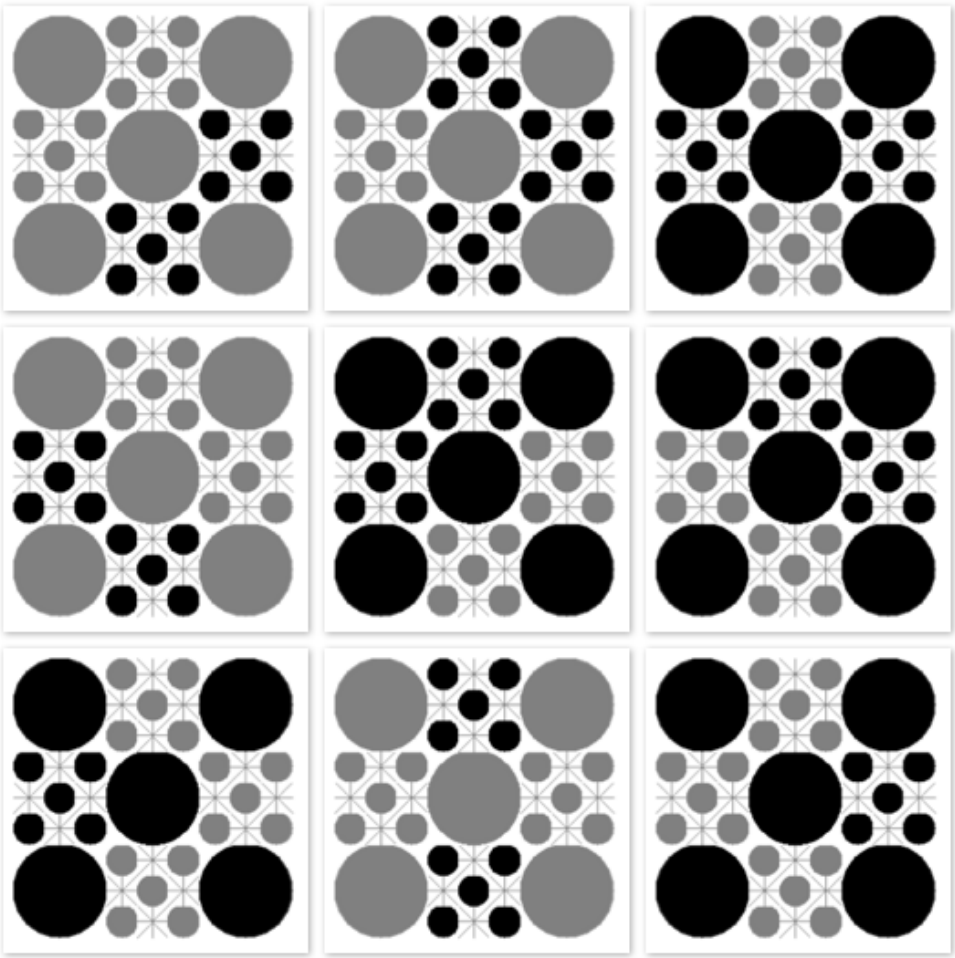


Figure D.5: Example of fourth password selection screen