UNIVERSIDAD NACIONAL DE COLOMBIA

Human Interaction Proofs Based on Emerging and Multistable Images: A Practical Application of the Theory of Representation of Algebras

María Alejandra Osorio Angarita

Universidad Nacional de Colombia
Facultad de Ciencias, Departamento de Matemáticas
Bogotá, Colombia
2019

Human Interaction Proofs Based on Emerging and Multistable Images: A Practical Application of the Theory of Representation of Algebras

María Alejandra Osorio Angarita

Tesis de grado presentada como requisito parcial para optar al título de:
PhD en Ciencias Matemáticas

Director:
Ph.D. Ebroul Izquierdo

Codirector:
Ph.D. Agustín Moreno Cañadas

Línea de Investigación:
Matemática Aplicada
Grupo de Investigación:
TERENUFIA-UNAL

Universidad Nacional de Colombia
Facultad de Ciencias, Departamento de Matemáticas
Bogotá, Colombia
2019

(Dedicatoria)

A Dios que me bendice, engrandece y fortalece, por permitir que los sueños y propósitos se hagan realidad.

A mi madre que desde el cielo me guía y acompaña.

A María Alejandra, Sebastián y Felipe que son el motor y grandeza de mi vida, por su apoyo, paciencia y confianza.

Agradecimientos

Inicialmente quiero presentar un agradecimiento muy especial a los profesores Ebroul Izquierdo y Agustín Moreno Cañadas, director y codirector de esta tesis, quienes con su gran conocimiento, experiencia y paciencia han orientado este trabajo de excelente forma, brindando ideas y aportes significativos a la investigación. A la Doctora Fiona Rivera, quien me brindó invaluables aportes en la redacción del documento. De igual manera expreso mi agradecimiento a COLCIENCIAS, por su apoyo financiero en parte de esta investigación.

Resumen

Se explora el uso de la Teoría de Representaciones de Álgebras para construir repositorios de imágenes emergentes, dichas imágenes se usan en diferentes tipos de pruebas interactivas con humanos. Estos tests pueden diferenciar humanos de bots o robots, con el fin de proteger ambientes en línea (tales como redes sociales, wikis, ventas de tiquetes, proveedores de correo gratis, etc), de diferentes tipos de amenazas de seguridad.

Se introducen algoritmos novedosos para modelar imágenes emergentes y multiestables a partir de órdenes tejados, configuraciones de Brauer, posets, junto con herramientas y técnicas que provienen del Análisis topológico de datos (nubes de puntos, complejos simpliciales, y triangulaciones del espacio, entre otras), con el fin de crear formas que puedan ser reconocibles por los humanos pero difíciles para las máquinas.

Palabras Clave: (Configuración de Brauer, CAPTCHA, Emergencia, Imágenes emergentes, Pruebas Interactivas con Humanos, Posets, Seguridad, Órdenes tejados, Teoría de Representaciones de Álgebras).

Abstract

We explore the use of the theory of representation of algebras to construct emerging image-repositories, such emerging images are used in different types of human interaction proofs (HIPs). These tests are able to tell apart human from bots (or robots) in order to protect online environments (as social networks, wikis, ticket sellers, free-email providers, etc) from different kind of security threats.

We introduce novel algorithms to model emerging and multistable images from tiled orders, Brauer configurations, posets together with tools and techniques arising from TDA (point clouds, simplicial complexes and spatial triangulations, among others), in order to create shapes which can be identified by humans as recognizable images hard to detect by machines.

Keywords: (Brauer configuration, CAPTCHA, Emergence, Emerging Images, Human Interaction Proofs, Posets, Security, Tiled Orders, Theory of Representation of Algebras).

Contents

List of Figures

List of Abbreviations

AI	....	Artificial Intelligence.
ATGEI	....	Algorithm to generate emerging images.
ATMMEI	....	Algorithm to model multistable and emerging images.
ATTACH	....	Algorithm to tell Apart Humans and Machines.
BC	....	Brauer Configuration.
BCA	....	Brauer Configuration Algebra.
CAPTCHA	....	Completely Automated Public Turing test to tell Computers and Human Apart.
DSA	....	Digit Sequencing Algorithm.
EEP	....	Emerging Equipped Poset .
EI	....	Emerging Image.
HIPs	....	Human interaction Proof.
HVS	....	Human Visual System.
JIM	....	Jigsaw Image Mosaic.
MGA	....	Mask Generating Algorithm.
OCR	....	Optical Character Recognition.
PIM	....	Puzzle Image Mosaic .
POSET	....	Partially Ordered Set.
REIADT	....	Reconstruction of Emerging Images by Restricted Admissible Transformations
RTT	....	Reverse Turing Test
TDA	....	Topological Data Analysis.
URL	....	Uniform Resource Locator.
VC	....	Visual Cryptography.
VLC	....	Variable Length Code.
VSS	....	Visual Secret Sharing.

VSSS	....	Visual Secret Sharing Schemes.
WWW	....	World Wide Web.

Lists of Symbols

Representation Theory

$\mathcal{P}$	—	Poset .
$\mathbb{R}^2$	—	Euclidian Plane.
C	—	Chain.
$a^{\vee}$	—	Ordinary upper cone.
$a_{\wedge}$	—	Ordinary lower cone.
$a^{\triangledown}$	—	Strong upper cone.
$a_{\vartriangle}$	—	Strong lower cone.
$a^{\blacktriangledown}$	—	Truncated upper cone.
$a_{\blacktriangle}$	—	Truncated lower cone.
M	—	Matrix presentation.
U	—	Representation of a poset.

Tiled orders

T	—	Field.
$\mathbb{O}$	—	The normalization ring.
$\mathbb{Z}$	—	The set of Integers
$\mathcal{L}$	—	Lattice.

Emerging Images via Tiled Orders

X_k	—	Binary Image.
X'	—	Image.
$a(X)$	—	Bitmap

B_i	—	Blocks.
m_i	—	Arrows.
n_i	—	Columns.
σ	—	Path.
m_{i_1,J_1}	—	Marker.
S	—	Set of Permissible Paths .
$\|\sigma\|$	—	Length of Path.

Brauer configurations

Γ_0	—	Vertex set.
Γ_1	—	Set of Polygons.
μ	—	Multiplicity Function.
$\mathfrak{o}$	—	Orientation of Γ.
Λ_Γ	—	Brauer configuration algebra.
Q_Γ	—	Quiver of a Brauer configuration.
$\mathcal{V}, \mathcal{W}$	—	Polygons.
v_i	—	Vertices.

Homological Persistence

I	—	Image.
$\mathbb{X}$	—	Sample of topological space.
S	—	Sample of [1,m]x[1,n].
m	—	Number of arrows.
n	—	Number of columns.
A	—	Point cloud.
K	—	Simplicial Complex.
p	—	Number of points.
ϵ	—	Radius of balls.

Introduction

In recent years, many applications of the theories of representations of posets and algebras have emerged. These applications have allowed advances in areas such as Number Theory, Combinatorics, Information Security and Big Data Analysis, among others.

Representation theory is the study of the ways in which a given group may act on vector spaces, so that, it studies the way of representing elements of algebraic structures via linear transformations between vector spaces. It allows to find relationships that expresses similarities between objects. In this thesis we study the phenomenon of emergence from different points of view of the theory of representation of algebras.

The term emergence generally refers to the process of becoming visible after being concealed. In the context of images, emergence is the phenomenon by which the object of the image is imperceivable through recognition of the individual elements. Instead, the object is, only perceivable when viewed as a whole, i.e. all at once. In small local neighbourhoods the elements of the image look complex, random and therefore meaningless. However, when observed in its entirety, the elements are aggregated and the main subject in the image suddenly pops out, i.e. emerge and is thus perceived as a whole. Thus, Emerging Images (EI) are images with the property of Emergence; this property has been well studied by the Gestalt School [38].

The exact process of how such objects within emerging images are perceived is currently unknown and thus it can be concluded that it is extremely challenging, if not impossible, to automate the recognition process [64]. For this reason, emergence can be leveraged to make Human Interaction Proofs (HIPs), i.e. systems with the main objective of distinguishing between various groups of users through a challenge/response protocol. HIPs protocols can be leveraged to distinguish human versus machine, or one person versus another, etc [12].

In 2009, Mitra, et al. proposed a synthesis technique to generate emerging images of 3D objects that are detectable by humans, but are difficult to recognize by computer vision algorithms [64].

Informal Human Interactive Proofs were introduced by Naor (1996), who proposed using a “Turing Test” to verify that queries to a service over the web are being made by a human being rather than a machine [67]. Subsequently, Von Ahn (2000) coined the term CAPTCHA (Completely Automated Public Test to Tell Computers and Humans Apart). More specifically, CAPTCHAs aim to discriminate between actions executed by computers and those executed by humans [1].

Thus the primary aim of this research is to introduce the concept of Emergency from different mathematical structures. This fact is very relevant because it is a concept that it only had been studied by the psychologists.

Taking into account the above, it is necessary to propose the generation of emerging images in a way that it is not easy for the computational algorithms to decipher the information. For this reason, this thesis aims to create emerging images from mathematical concepts and structures, such as semimaximal rings known as tiled orders, posets, Brauer configurations and Topological Data Analysis. Additionally, we present connections between emerging images and these mathematical structures. The practical application of these results consists of using them in human interaction tests or proofs.

On the other hand, modeling means finding a mathematical representation for an object, a process or a system not mathematical, constructing a theory or mathematical structure that incorporates its essential features. The built mathematical model, allows to obtain results about the process in question. Currently, models are simulated in computers in order to predict results and to contribute in the resolution of problems in other knowledge areas. In this thesis a novel mathematical model for generating multistable and emerging images is presented.

Furthermore, the presentation of novel algorithms for generating multistable and emerging images that are constructed via 01-Tiled Orders, posets, Brauer configuration and homological persistence. This kinds of images may be used to construct different and interesting Human Interaction Proofs.

0.1 Contributions

The main contribution of this thesis is:

“A key security application of the Theory of Algebra Representation”.

We introduce a novel strategy to produce HIPs exploiting tools of theory of algebra representation . The proposed approach is based on the generation of images of shapes, that can be identified by humans but hard to be recognised by a computer program. Several experimental results are reported to demonstrate the robustness and feasibility of the proposed approach.

In detail, the main contributions of this thesis are the following.

- Definition of a Brauer configuration associated to emerging images.
- Design and development of an efficient digits sequencing algorithm (DSA).
- Schemes associated to some Visual Secret Sharing.
- A mathematical model for emerging and multistable Images.
- The introduction of length-variable error-correcting codes based on Brauer configuration algebras and quivers.
- Design of an efficient algorithm to model multistable and emerging images ATMMEI [26].
- Design and development of algorithm (ATGEI) to generate emerging images via tiled orders.
- Definition of Emerging Equipped Posets.
- Definition of a Brauer configuration associated to multistable images.
- Design of an efficient algorithm (MGA) to generate random masks from topological techniques.

- Comparison between homological methods and Tiled orders methods to generate emerging images.

- Human Interactive Proof: EmerCAPTCHA and experiment applied to users.

- Algorithm to tell Apart Humans and Machines (ATTACH).

- Human Interactive Proof based on images obtained from Algorithms MGA and ATTACH and experiment applied to users.

- Software called System REIADT.

- Repository of emerging and multistable images.

- Talk: "Matrix Problems to Generate Mosaic-Based CAPTCHAs", ICDP 2015. London (UK).

- Poster: "Emerging Images-based CAPTCHAS". MATH AMSUD. 2016. Montevideo (Uruguay).

- Talk: "Algorithms of Differentiation and Its Applications", 2017, CLA Quito-Ecuador.

- Talk: " Matrix Problems Associated to Some BCA and Its Applications", CLA , México D.F. 2019

- Talk: "Human Interaction Proofs (HIPs) based on Emerging Images and Topological Data Analysis (TDA) Techniques", 3rd Cyber Security and Networking 2019. Quito (Ecuador).

- Paper published: "Matrix Problems to Generate Mosaic-Based CAPTCHAs" [26].

- Paper accepted: "Human Interaction Proofs (HIPs) based on Emerging Images and Topological Data Analysis (TDA) Techniques" [5].

- Paper accepted: "Human Interaction Proofs (HIPs) Based on Multistable Images and Brauer Configuration Algebras (BCA)" [6].

- Paper submitted: "Algebraic Tools for Multimedia Based Cryptography and Security Applications" [7].

- Paper submitted: "Brauer Configuration Algebras for Multimedia Based Cryptography and Security Applications" [8].

0.2 Overview of the thesis

This thesis consists of eight chapters. Following the introductory chapter, the Chapter 1 deals with the main concepts, principles and theoretical results of Gestalt Theory. An overview of Human Interactive Proofs and CAPTCHAs, background and related work about them are showed. Furthermore, definitions about emerging images, Representation of algebras, Brauer configuration algebras, representation of posets, tiled orders and homological persistence are presented. And the basic concepts regarding visual cryptography schemes are presented. At the end of this chapter, we present a first application of the Brauer configuration algebras in the construction of length-variable error-correcting codes.

In the Chapter 2 a Brauer configuration associated to emerging images is presented, which allows to generate schemes from sequencing process of digits. It also presents the Brauer configuration induced by some exchange rules and schemes of color VSSS defined by suitable Brauer configurations.

The Chapter 3 deals with a novel mathematical model and its algorithm to generate emerging and multistable images.

The Chapter 4 shows emerging images associated to tiled orders and posets, an algorithm for generating emerging images is also presented.

In the Chapter 5 deals with a Brauer configuration associated to multistable and emerging images.

Chapter 6 focuses on the using of theoretical concepts, tools and techniques arising from Topological Data Analysis (points clouds, simplicial complexes and spatial triangulation among others), in order to create random masks which are used to generate emerging images.

In Chapter 7 an algorithm to tell humans and machines is introduced. In addition, we describe experimental and statistical results. Additionally, it presents both the experiments made to evaluate the usability, and the human friendliness and the results for these experiments. Screenshots about System REIADT are showed, which allows to obtain computational results.

The conclusions and recommendations for future works are summarised in Chapter 8. Finally, appendices and the list of author's publications is given at the thesis end along with the references used.

Chapter 1

Preliminaries

In this chapter, we present some theories and basic definitions which will be used in the thesis. We start discussing some aspects regarding the understanding human psychology and the visual perception which are very important factors to create successful Human Interaction Proofs (HIPs).

1.1 Gestalt Theory, Principles and Laws

Gestalt theory was developed in Germany by Max Wertheimer, Wolfgang K*ö*hler, and Kurt Koffka in 1923 [44], Wertheimer et al. believed that people perceive a structure of components that they treat as a whole. This theory arose in 1890 as a reaction to the psychological theory of the time named atomism. Atomists believed the nature of things to be absolute and not dependent on context [38].

The concept of **gestalt**, (the German word for "essence or shape of an entity's complete form"), was first introduced to contemporary psychology by Christian Von Ehrenfels following the theories of David Hume, Johann Wolfgang Von Goethe, Immanuel Kant, David Hartley, Ernst Mach and Max Wertheimer [44]. Subsequently, Fritz Perls, Laura Perls and Paul Goodman created gestalt therapy by bringing together the diverse European and American theories and backgrounds to synthesise a new psychotherapy and social theory [20].

The basic principle of the Gestalt school is: " The whole is other than the sum of the parts" [60], i.e. the properties of the totality do not result from the constituent elements. Instead they emerge from the temporal space relations of the whole and the human eye sees objects in their entirety before perceiving their individual parts. Furthermore, the gestalt effect stipulates that perception is the product of complex interactions among various stimuli, and it depends on the form-generating capability of our senses to perceive whole forms instead of a mere collection of simple lines and curves [56].

A. Desolneaux et al. (2006) described the Gestalt theory as a substantial scientific attempt to state the laws of visual reconstruction [38]. In the Wertheimer programme there are two kinds of organizing laws: grouping laws, and those related to governance and conflict [38].

Grouping laws, starting from the atomic local level, recursively construct larger groups in the perceived image. Each grouping law focuses on a single quality (e.g. colour, shape, or direction, . . .).

In the cases of principles governing the collaboration and conflicts of Gestalt laws, groups are identifiable with subsets of the retina. In image analysis we identify them with the points of the digital image. Whenever points (or previously formed groups) have one or several characteristics in common, they get grouped and form a new larger visual object: a *gestalt*.

The list of elementary grouping laws given by Gaetano Kanizsa in *Grammatica del vedere* (1980) is *vicinanza, somiglianza, continuita di direzione, completamento anodale, chiusura, largheza constante, tendenza alla convessita, simetria, movimento solidale, esperienza passata* [57]; i.e. vicinity, similarity, continuity of direction, amodal completion, closure, constant width, tendency to convexity, symmetry, common motion, past experience. This list is very close to the list of grouping laws considered in the founding paper by Wertheimer [44].

At the beginning of the 20th century, the school of Gestalt practised a series of theoretical and methodological principles that attempted to represent the subjective experience of perception [60, 62], such as the following:

- **Principle of totality**: the conscious experience should be viewed holistically, as a totality of the dynamic interactions of components of the brain.

- **Principle of psychological isomorphism**: there is a correlation between the perceptual phenomena and the activity in the brain.
- **Phenomenon experimental analysis**: any psychological experiment should have as a starting point a phenomena and not sensory qualities.
- **Biotic experiment**: stats the need of conducting real experiments on natural situations and real conditions to reproduce with higher fidelity the habitual situations of the subjects.

The Gestalt theorists have used four key gestalt properties to describe the processes of visual perception: Emergence, Reification, Multistability and Invariance [38,57,62,65,92]. The ubiquity of these properties in every aspect of perception suggests that gestalt phenomena are fundamental to the nature of the perceptual mechanism [62].

1.2 Key Gestalt properties

1.2.1 Emergence

Emergence is the process of complex pattern formation from simpler rules. The main characteristic is that the final global form is not computed in a single pass but continuously. An example can be observed in Figure 1.1. The local regions of the image do not contain enough information to distinguish significant form contours from insignificant noisy edges, but as soon as the subject is recognised, the perception of a dog is very vivid despite the fact that much of its perimeter is missing. The Gestalt theory does not offer any specific computational mechanism to explain this property in visual perception [62].

1.2.2 Reification

Reification is the constructive or generative principle of perceptual processing, by which the final form is perceived by filling-in of a more complete and explicit perceptual entity based on a less complete visual input. The Kanizsa's figures shown in Figure 1.2 are of the most well known illusions produced by the Gestalt theory. For example, in the figure, a triangle can

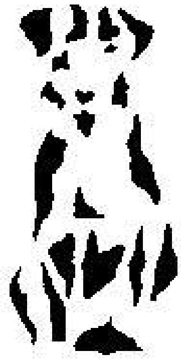


Figure 1.1: Example of the Gestalt property Emergence: Dalmatian dog. Google's Repositories.

be recognised by filling-in perceptually, and producing visual edges in places where there are none in the input [62].

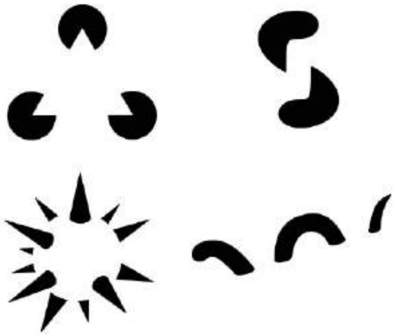


Figure 1.2: Example of the Gestalt property: Reification showing the Kanizsa's images. Google's Repositories.

1.2.3 Multistability

Multistability refers to the visual process of perception. Perception must involve some kind of dynamic process which stable states represent the final percept [62]. According to Sterzer et al. (2009) *multistable perception* is the spontaneous alternation between two or more perceptual states that occur when sensory information is ambiguous [86].

As an example of multistability, the well known Old-woman/Young-woman drawing which consists of a single set of lines is presented in Figure 1.3. This single image involve ambiguity, so that the drawing can be visually interpreted in at least two ways that are mutually exclusive, depending on how the patterns or structures of the eyes interrelate those lines.

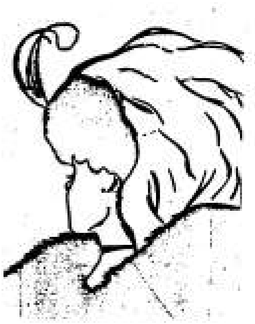


Figure 1.3: Example of the Gestalt property Multistability: Old woman and young woman. Google's Repositories.

The Figures 1.4 and 1.5 show other examples of images where it is possible to perceive multistability property.

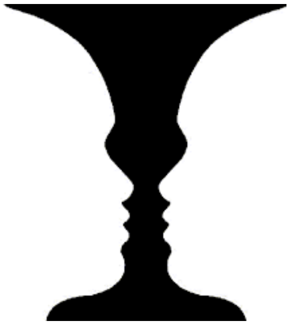


Figure 1.4: Multistability Example: Symmetry. Google's Repositories

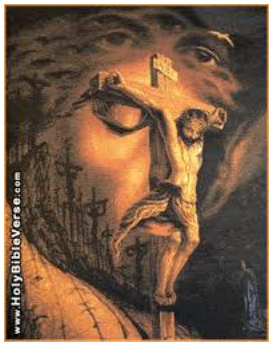


Figure 1.5: Example of Multistability: Jesus. Google's Repositories.

1.2.4 Invariance

Invariance is the property by which an object can be recognised regardless of its rotation, translation, scale, change of lighting or background, or texture and motion [62]. See Figure 1.6.

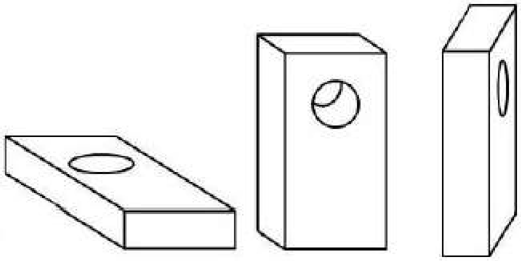


Figure 1.6: Example of the Gestalt property Invariance. Google's Repositories.

1.3 Gestalt laws

The Gestalt principles of perception come from the Law of Pragnanz, the German word for language. The principles describe the organization of perceptual scenes. The Law of Pragnanz says that when we look at the world we usually perceive complex scenes composed of many groups of objects on some background, with the objects themselves consisting of parts, which may be composed of smaller parts, etc. and we tend to order our experience in a manner that is regular, orderly, symmetric, and simple. The interpretations of sensation that comes from the perception are called the 'Gestalt Laws' [44] which are presented below.

- **Law of Proximity**: Objects that are closer together are perceived as more related than objects that are further apart, i.e. elements tend to be perceived as aggregated into groups if they are near each other. For example, in Figure 1.7, the first row is perceived as a sextuplet while the second and third row, due to the change of distance between some of the components. The patches are perceived not just collectively as a sextuplet, but also as being subdivided into groups.

  The objects do not need to be similar in any other way beyond being grouped near each other in space in order to be seen as having a proximity relationship.

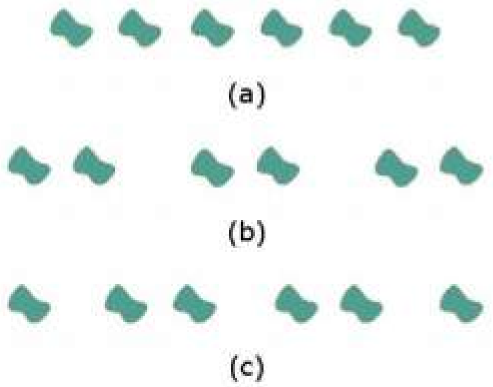


Figure 1.7: Law of Proximity Example. [77].

- **Law of Similarity**: Elements tend to be integrated into groups if they are similar to each other. This similarity can occur in the form of shape, colour, shading or other qualities. For example, as shown in Figure 1.8, the shapes have a constant distance between them, but they are perceptually partitioned into three adjacent pairs; due to the similarity of visual attributes.

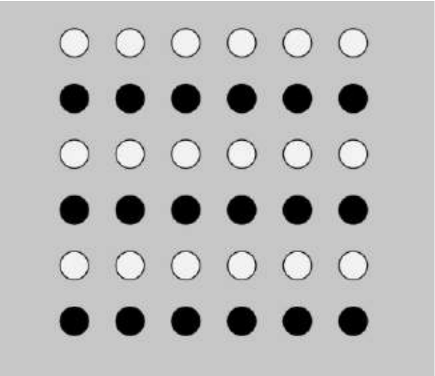


Figure 1.8: Law of Similarity Example. Google's Repositories.

- **Law of Symmetry**: The mind perceives objects as being symmetrical and forming around a centre point. When two symmetrical elements are unconnected the mind perceptually connects them to form a coherent shape. Similarities between symmetrical objects increase the likelihood that objects will be grouped to form a combined symmetrical object.

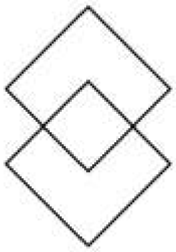


Figure 1.9: Law of Symmetry Example.

- **Law of Common Fate**: Elements tend to be perceived as grouped together if they move together. We perceive elements of objects to have trends of motion, which indicate the path that the object is on. For example, if there is a line of dots, and half the dots are moving upwards while the other half are moving down, we would perceive the upward moving dots and the downward moving ones as two distinct units.

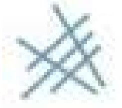


Figure 1.10: Law of Common Fate Example. Google's Repositories.

- **Law of Continuity**: Oriented elements or groups tend to be integrated into perceptual wholes if they are aligned with each other. In cases where there is an intersection between objects, individuals tend to perceive the two objects as two single uninterrupted entities. Stimuli remain distinct even with overlap. We are less likely to group elements with sharp abrupt directional changes as being one object. Elements arranged on a line or curve are perceived as more related than those in neither case. An example is presented in Figure 1.11.

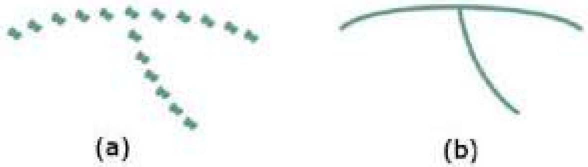


Figure 1.11: Law of Continuity Example [77].

- **Law of Closure**: Elements tend to be grouped together if they seem to complete some entity. More specifically, when parts of a whole picture are missing, our perception fills in the visual gap. In the Figure 1.12, you probably see the shapes of a circle and rectangle because your brain fills in the missing gaps in order to create a meaningful image.

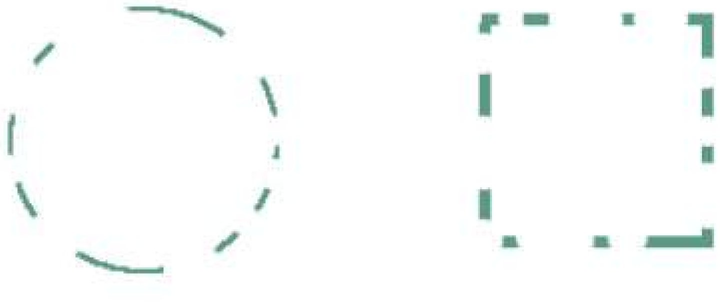


Figure 1.12: Law of Closure Example [77].

- **Law of Pragnanz**: The word *pragnanz* is a German term meaning "good figure". The law of Pragnanz is sometimes referred to as the Law of Good Figure or the Law of Simplicity. This law holds that objects in the environment are seen in a way that makes them appear as simple as possible. In the Figure 1.13, look more the images as a series of overlapping circles rather than an assorment of curved, connected lines.

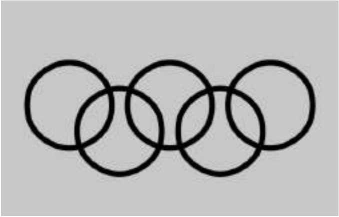


Figure 1.13: Law of Pragnanz Example.

- **Law of Good Gestalt**: Elements tend to be grouped together if they are parts of a pattern which is a good Gestalt, i.e. given the input as simple, orderly, balanced, unified, coherent, regular, etc. as possible. This law implies that as individuals perceive the world, they eliminate complexity and unfamiliarity in order to observe a reality in its most simplistic form. The Law of Good Gestalt focuses on the idea of conciseness which is what all Gestalt theory is based on. See Figure 1.14.

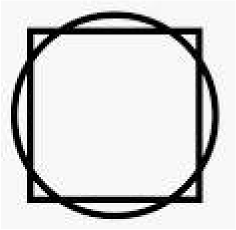


Figure 1.14: Law of Good Gestalt Example.

- **Law of Past Experience**: Elements tend to be grouped together if they were together often in the past experience of the observer. If two objects tend to be observed within close proximity, or small temporal intervals, the objects are more likely to be perceived together.

1.4 Human Interaction Proofs: Background and Related Work

The rapid evolution of the internet, the World Wide Web (WWW) and its use in all business sectors and aspects of life (such as education, web search, emails, the purchase of goods or services), has led to the need to create security mechanisms to avoid threats and infractions. Internet security has been an important issue since the internet's arrival in the 1980s [12]. For example, websites that carry commercial or administrative applications commonly require forms to be filled out to enable authorised people to use their services. However, some users abuse these services by creating programs called bots, (an acronym of robot), to register automatically to use the services for malicious purposes [76].

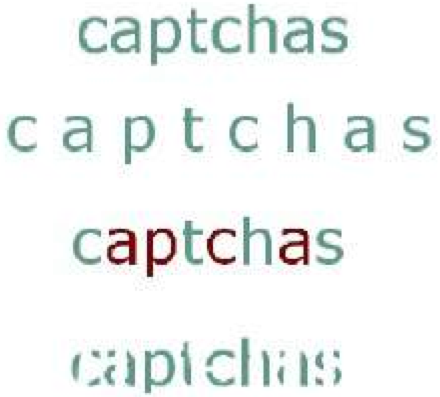


Figure 1.15: Law of Past Experience Example [77].

An internet bot, in its most generic sense, is software that performs an automated task over the internet. More specifically, a bot is an automated application used to perform simple and repetitive tasks that would be time-consuming, mundane or impossible for a human to perform. Although bots can be used for productive tasks, they are frequently used for fraudulent activities [89].

Malicious bots are typically blended threats that are part virus/worm, part bot and are used in identity theft or to launch attacks for denial of services. They can also generate spam, i.e. the sending of unsolicited email messages, including for the purpose of identity theft or financial fraud. In addition, spam can be used to attack personal computers through viruses, Trojan horses or malicious software [12]. Other illegal bot uses include harvesting email addresses for spam, scraping content, or manipulating comments/votes on sites that allow user feedback [89].

In response to increasing threats, interactive tests have thus been proposed to distinguish human users from automated processes. Such tests facilitated the websites in providing security to the user when using their services.

This chapter presents the state of the art and the different strategies used based on the HIPs and CAPTCHA Methodology. Furthermore, the concept of Emerging Images is presented because they are the study object in this thesis.

1.4.1 Human Interaction Proofs

A HIP, or Human Interaction Proof or Human Interactive Proof, is a human authentication mechanism that generates and grades tests to determine whether the user is a human or a malicious computer program [69].

Diverse systems of HIPs have been presented over recent years with their main objective to distinguish between various groups of users through a challenge/response protocol, e.g., human versus a machine, one person versus anyone else, etc [12].

Some researchers present Human Interaction Proofs, looking for an ideal system that cannot be passed by the computers and other researchers look for to create a system that can pass the test. Consequently, this situation makes that HIPs also become difficult for potential users too. So there, is a tradeoff between the usability and robustness in designing HIP tests.

The concept behind HIP tests comes from a methodology proposed by Alan Turing, which tests the intelligence of a computer through an 'imitation game'. In the Turing test, a human judge asks questions to a human and a computer (situated in different rooms). If the interrogator cannot determine which room the computer is in, and which one the human, the computer has passed the Turing test [93].

Naor (1996) proposed using a Turing Test for the purpose of verifying that a human being is making a query to a service over the web, rather than an automated process. Thus, before a request is processed, the user should be presented with an instance of a chosen problem as a challenge easy for humans to solve, but one that the best known programs fail on a non-negligible fraction of the instances [67].

S. Shirali and M. Shirali studied available HIP systems, various applications of HIP systems and attacks conducts on them [83].

In January 2002 ran the first workshop on HIPs, this indicate the importance of research in this field. In 2005 the second workshop was held and in its conclusions they suggest that the existing HIPs tried to maximize the difficulty for automated programs to pass tests by increasing distortion or noise, consequently it has also become difficult for potential users too [12].

Therefore, some aspects that should be in the design and construction of HIPs are the usability and robustness of tests.

For instance, ARTiFACIAL [78] works as follows: per each user request, it automatically synthesizes an image with a distorted face embedded in a cluttered background. The user is asked to first find the face and then click on 6 points (4 eyes corners and 2 mouth corners) on the face. If the user can correctly identify these points, it concludes the user is a human; otherwise, the user is a machine, see Figure 1.16.

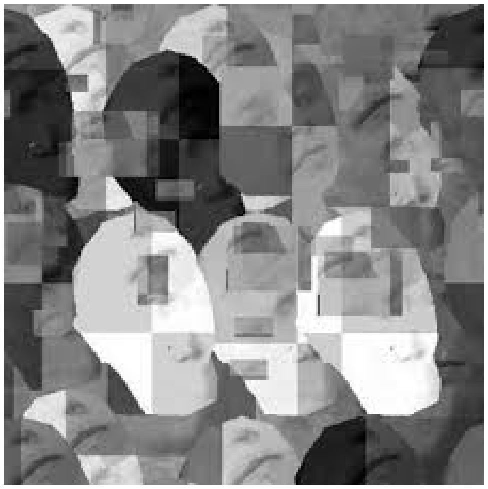


Figure 1.16: Images of ARTiFACIAL [78].

Chew and Baird (2003) [35] proposed BaffleText, a HIP which uses non-English 'pronounceable words' to defend against dictionary attacks, and Gestalt-motivated image-masking degradations to defend against image restoration attacks, that exploits the difference in ability between humans and machines in reading images of text. They constructed website and invited 33 employers to visit and transcribe texts. Their experiments confirmed the legibility and user acceptance of BaffleText Images.

Chellapilla presented two works where the friendly aspect is the most important one [33, 34].

The evolution of the Human Interaction Proofs and its applications are presented by Baird et al. (2005) [12].

Some authors use the terms: Human Interactive/Interaction Proofs (HIPs), Reverse Turing Tests (RTTs) and CAPTCHAs indistinctly, but others establish important differences. Chew [35] claims that a CAPTCHA is a kind of HIP, but the first ejects an automatic evaluation. That is, CAPTCHA is a type of HIP with additional features.

Ugochukwu et al. (2013) presented a review and evaluation of HIP in online environment, given their lack of acceptance of users [71]. Furthermore, they propose to use biometric authentication in conjunction with HIP, in order to obtain better results.

Within the evolution of the Human Interactive Proofs, the most outstanding to differentiate a human from a computer are CAPTCHAs, which are described in the next section.

1.4.2 CAPTCHAs

The term CAPTCHA stands for Completely Automated Public Turing test to tell Computers and Human Apart. It is perhaps the most expanded branch of the HIP systems. A CAPTCHA is a software that generates graded tests that most humans can pass but computers cannot. The origins derive from 1997 when Altavista developed a filter that generated images of printed random characters to avoid automatic submission of Uniform Resource Locators (URLs) to their search engine. Subsequently, Blum et al. (2000) created the CAPTCHA project which was developed at Carnegie Mellon University [18]. Blum et al. articulated the most desirable properties a CAPTCHA test should have:

- the test's challenges should be automatically generated and graded (the judge is a machine)

- the test should be taken quickly and easily by human users

- the test should accept virtually all human users and will reject virtually all machine users

- the test should resist automatic attack for many years in spite of technology advances or open test algorithms

Von Ahn et al. (2003) defined formally a CAPTCHA, as a test C, (α, β)-executable in the sense that at least a portion α of population has success greater than β over C [1].

The test developed by Von Ahn in 2003 gives the user two words: one where the answer is unknown and a second control word for which the answer is known. If users correctly type the control word the system assumes that they are humans [1]. Such a program can be used to distinguish humans from computers and has many applications for practical security to prevent unauthorised access to: online polls, free email services, search engine bots, and prevent worms and spam and dictionary attack for example.

A second definition: CAPTCHA is a cryptographic protocol whose underlying hardness assumption is based on an Artificial Intelligence problem (AI) [1].

The essential exploitable concept of leveraging CAPTCHA is thus that most humans can pass such tests but computers algorithms cannot [18]. CAPTCHA programs can therefore protect internet companies and human users against spam or bots, through the generation of graded tests. CAPTCHAs are thus used to protect websites and services, including free email providers, ticket sellers, social networks, wikis and blogs due to their capabilities of distinguishing human users from automated processes [2,3].

Following a concept introduced by Von Ahn et al. (2003) [2], for a CAPTCHA to be considered good currently, it should have the property of being associated with a hard Artificial Intelligence (AI) problem. Calling a hard AI reflects that it would not be solved by simple specific algorithm, i.e. it is difficult and complex for machines. This is in the sense that correct solutions of a given CAPTCHA should only be attainable if the underlying AI problem is solved, Von Ahn et al. define this characteristic as a win-win situation: either the CAPTCHA is not broken and there is a way to differentiate humans from computers, or the CAPTCHA is broken and a useful AI problem is solved [2].

Regardless of the type of method used in the CAPTCHAs, they share common characteristics that define them. First, the generation of the tests should be completely automated by a machine. Only human intervention should be required to pass the test. Second, the code, the data, and the algorithm should ideally be public since CAPTCHAs benefit from peer review, which is normally successful at identifying weaknesses [66]. Finally, a robust CAPTCHA should rely on a completely random generation system for choosing the corresponding characters, images or other files. The solutions should not be contained in databases because they could be cracked. Additionally, the machine generating the tests should not be able to solve them. The aim is to create a CAPTCHA that is immune to imminent attacks.

The primary application of CAPTCHA is to prevent malicious attacks to the systems by spammers. However, they also serve to protect vulnerable systems, such as Yahoo or Hotmail, against e-mail spam, automated posting to forums, blogs and wikis as a result of commercial interests or harassment. Another important function is bit rate limiting when excessive use of a service is observed.

Many websites, including well known players such as Google, Gmail and Yahoo mail, use CAPTCHAs to facilitate barring unauthorized access to users accounts. CAPTCHAs are also commonly used in sites that provide access to confidential information, such as bank accounts, credit cards, and websites that take in payments for services or goods.

CAPTCHAs use for websites aids securely saving and protecting user data. CAPTCHAs offer protection against remote unauthorised digital inputs, since they facilitate ensuring that only a human being with the correct password can access, rather than an automated process designed to crack a password or example.

CAPTCHA tests were designed to prevent fake registrations by computer programs in websites [12], but the number of applications has increased since then. Nowadays they are used to prevent email from worms and spam such as Kartaltepe and Xi propose in [58], [81]. In addition to the email spam problem, CAPTCHA tests are used to prevent fraud in online polls [1], search engine bots reading web pages [4], bots playing online games [53], [98] and dictionary attacks [31]. They are also used for detecting phising attacks [32], [79] or user authentication [46], [81, 82]. In the last decade, different type of methods have been developed to produce CAPTCHAs. In 2013, Romero proposed a detailed classification that starts with three main branches and divides into sub-categories [75].

In 2015 Reynaga in [73] claims most existing CAPTCHAs do not propertly fit mobile devices which forces the user to abandon their tasks. They studied the usability of four tests and identified design strategies for the development of new CAPTCHA for smartphones.

Although nowadays there are several categories of CAPTCHAs including: Text-based, Audio-based and Images-based, we will only focus in the first and the last categories.

Text-based CAPTCHA

In a word-based CAPTCHA, the characters are distorted to make its recognition more difficult for the bots. Among the basic distortions, it can use translation, rotation (clockwise or counterclockwise) and scaling, among others such as sight angle, lighting effects, context, and camouflage [33]. A word-based CAPTCHA test consists on an image that contains distorted and noisy characters or words. To solve this test, the user has to type the characters presented in the image. Usually, the distortions applied to the image are complicated enough to prevent a robot to recognise the word while allowing humans to do so. Applications can be appreciated in Figure 1.17.

Von Ahn et al. proposed in 2008 reCAPTCHA. It was originally conceived as a test based on the inability of some OCR-algorithms to identify distorted text [3].

Perhaps a method similar to reCAPTCHA can be used to annotate or tag large quantities of images. See Figure 1.18.

Image-based CAPTCHA

An image-based CAPTCHA contains primarily an image that the user has to recognise. Amongst these tests, the user can be asked to implement different kinds of actions; solve a quiz, match symbols, recognise faces, etc. Usually, the images do not appear straightforwardly, instead they can contain warping, occlusion or lighting effects to avoid being recognised by machines. The last type is a sound-based CAPTCHA, which was implemented in the first place for those users that cannot solve visual CAPTCHAs due to an

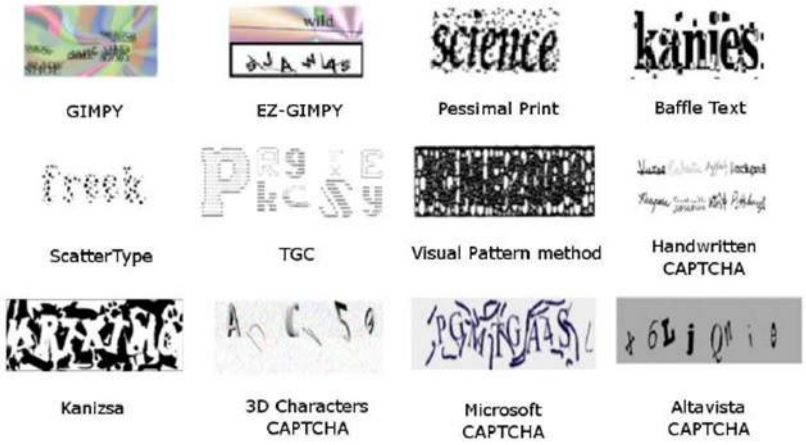


Figure 1.17: Image of Text-based CAPTCHA. Google's Repositories.

impairment. The test presents an audio file that contains words, letters, or numbers, mixed with background noise, that the user has to type correctly.

In 2013, a categorization of the available CAPTCHAs was presented by Romero, according to their characteristics, difficulties and friendliness for the user [77].

The three main groups are: OCR-Based, Visual Non-OCR-Based and Non Visual. OCR stands for Optical Character Recognition and it is an artificial intelligence program that is used for automatically reading scanned images of handwritten, typewritten or printed text. Normally, they are calibrated to recognise some specific character fonts and have difficulties when the image has low resolution. The recognition rate drastically drops at recognising cursive text, with recognition rates even lower than those of hand-printed text. The disadvantages of the OCR systems can be used as an advantage if applied to CAPTCHAs, so only human beings can recognise the text [33,34]. Nowadays, most of the methods to discriminate humans from computers are based on optical character or image recognition or sound recognition.

For instance, Asirra is a test that presents cats and dogs. See Figure 1.19.

Figure 1.18: Image of reCAPTCHA. Google's Repositories

S. Woo in 2019 [95] presented CAPTCHA characters on 3D objects. In this work, they exploit the difficulty that machines have in rotating 3D objects to find the correct viewpoint and in further recognizing characters in 3D. Participants agreed that their approach was usable in spite of the extra time required for 3D model rotation.

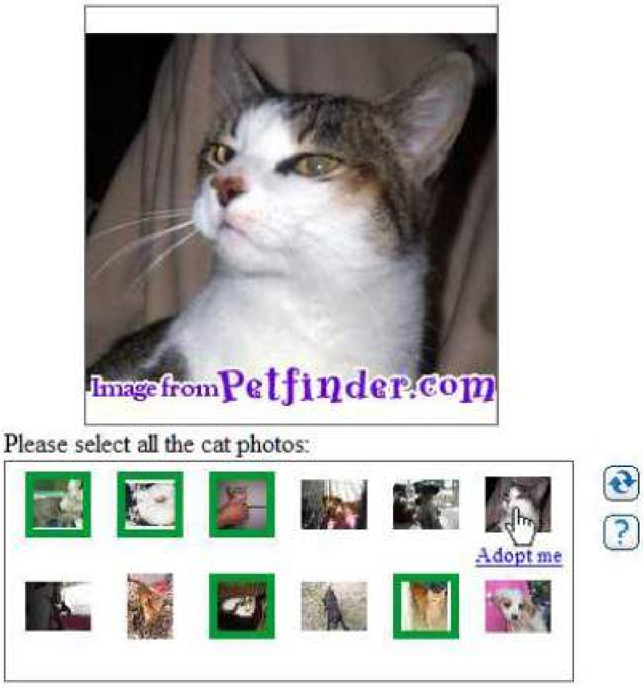
1.5 Emerging Images (EI)

Images with emergent properties illustrate one of the main ideas of the Gestalt School; Emergence is the phenomenon by which we perceive objects in an image not by recognizing the object's parts, but as a whole, i.e. all at once.

N.J. Mitra et al. (2009) proposed a synthesis technique to generate emerging images of 3D objects. Such images are gestalts with the property of to be detectable by humans but difficult to process by computer vision algorithms [64].

Figure 1.40 shows an example of the emergency property, on the left side the image a Dalmatian dog can be perceived, whilst on the right side the dog can be visualized more easily. Examples of images obtained by Mitra et al. are presented in the Figures 1.21 and 1.22.

Features mentioned above make the emerging images suitable for building Human Interaction Proofs (HIPs).



Please select all the cat photos:

Figure 1.19: Image of ASIRRA [95].

These ideas were used by Baird et al. in [13] to generate dynamic text strings-based CAPTCHAs. In this work, they also reported an attack to the famous NuCAPTCHA (see Figure 1.23) which has been considered the most secure and usable CAPTCHA. In fact, the test of Baird et al. is an improvement of NuCAPTCHA [13].

Remark 1.5.1. The main limitation with emerging images seems to be the difficulty to create a large amount of recognizable models.

Based on the notion of Emergence, Xu et al. (2013) developed the first concrete instantiation of emerging-image moving object called EIMO CAPTCHA

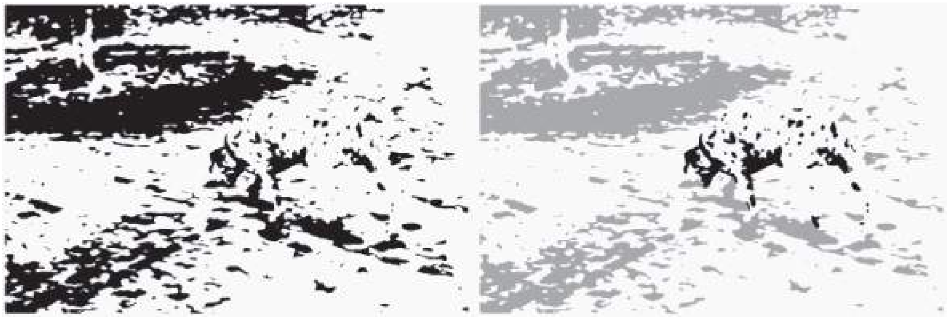


Figure 1.20: Emergence example: Dalmatian dog by R.C. James.

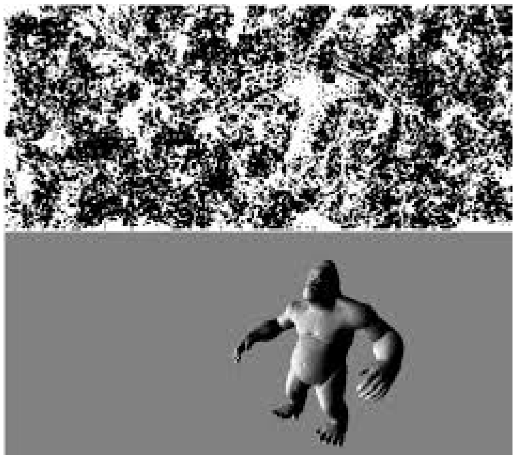


Figure 1.21: Emergence Example: Gorilla [64].

[96, 97], using 2D hollow objects (codewords), shown to be usable and to be secure.

Gao et al. (2015) introduced a new class of built CAPTCHA on the notions of Emerging Images and Dynamic Cognitive Games. Furthermore, they applied a series of countermeasures, such as pseudo 3D rotation, hidden edge segments, etc. to resist automated object recognition [49, 50]. Gao et al. (2017) showed the weakness of 2D EIMO CAPTCHA and they proposed a different design based on 3D objects and examined its security as well as usability [51, 103].

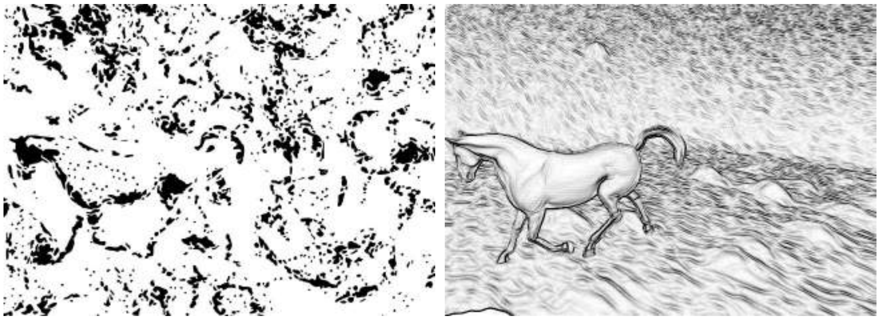


Figure 1.22: Emergence Example: Horse [64].

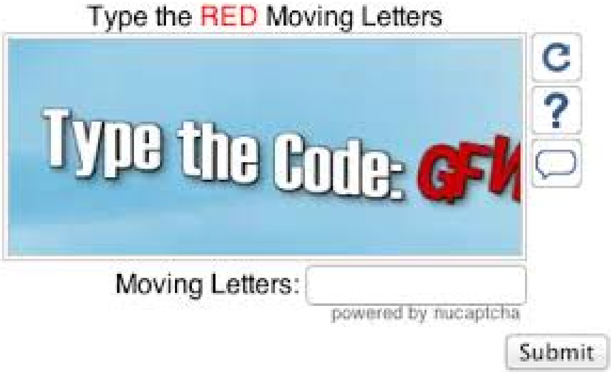


Figure 1.23: Example of moving-image object recognition from NuCaptcha [13].

Mosaic Images

Digital image mosaics are also built upon emergence. Mosaics are a form of art in which a large image is formed by a collection of small images called tiles.

Various mosaics can be created for an image depending of the choice of tiles and the restriction in their placement.

Tile mosaic, for example, are images made by cementing together uniformly coloured polygonal tiles carefully positioned to emphasize edges in the composite picture. Battiato et al. (2006) defined mosaics, in essence, as images obtained by cementing together small coloured fragments [14]. In Figure 1.24 one ancient mosaic, which was created manually, is shown; a modern mosaic image is shown in Figure 1.25. The latter was created with computational tools.

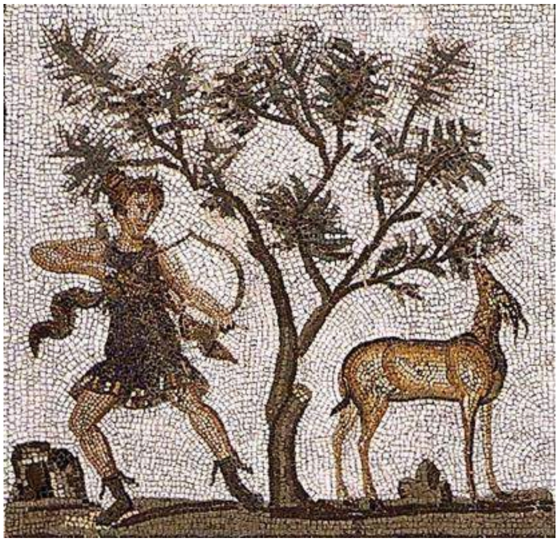


Figure 1.24: Ancient mosaic image example. Google's repositories

Mosaics are made of coloured tiles, called tessera or tesella, usually formed in the shape of a cube of materials separated by a joint of mortar. The mosaic surface exhibits irregular hollows (tesserae) and bumps (mortar) through the scene. Mosaics have numerous forms of irregularities, which is typical of the artwork style. Their shapes are irregular, from square shapes to polygonal ones [63].

Battiato et al. have worked in the mosaic images field with the purpose of reproducing the aesthetic essence of arts by means of computational tools [14,15].

Di Blasi et al. (2006) presented a technique to produce composite images called Puzzle Image Mosaic (PIM). Their method is inspired by Jigsaw Image Mosaic (JIM), where images tiles of arbitrary shape are used to compose the final picture. They proposed an algorithm that produces good results in lower time [17].

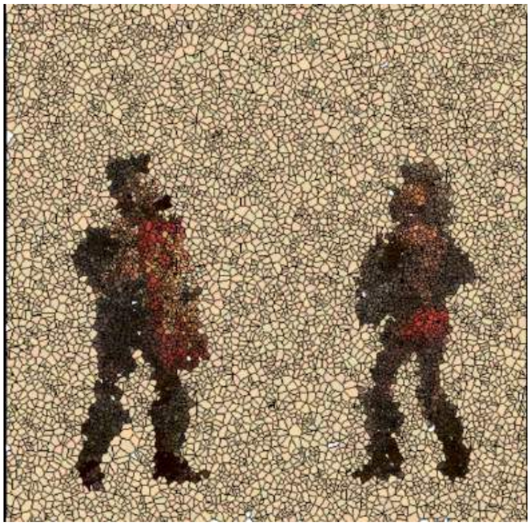


Figure 1.25: Example of modern mosaic image. Google's repositories.

Computational methods have also been proposed to generate mosaic images from Voronoi diagrams [39, 47].

S. Ming-Shing et al. (2007) aimed for Human Recognition of mosaic images that tends to be subjective, and an ideal mosaic image is unknowable. Human recognizes a mosaic image subjectively while computer vision algorithms measure a mosaic image objectively [63]. Therefore, it is possible to use mosaic images to make HIPs.

We must note that so far very little has been done in the investigation of the use of digital mosaic and multistable images to confuse bots.

Nested Images

A nested image is a form of artistic expression in which one or more secondary figures are embedded within a primary figure one by one. Contours of the primary figure, especially the contours of its inner holes, as used to portray a secondary figure. The secondary figure is totally inside the primary figure, and it would use some inner holes of primary figure as part of itself, which would produce and artistic effect [90, 91].

In 2017, Kuo et al. [61] proposed Generating Ambiguous Figure-Ground Im-

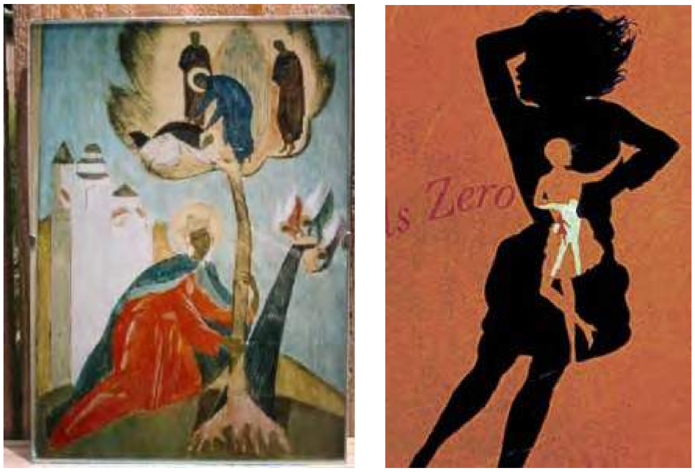


Figure 1.26: Examples of nested images. At right side, four ordered images, i.e., $b_1 \preceq b_2 \preceq b_3 \preceq b_4$ [90].

ages. Furthermore, they proposed an algorithm to match partially the content shape. See Figure 1.27.

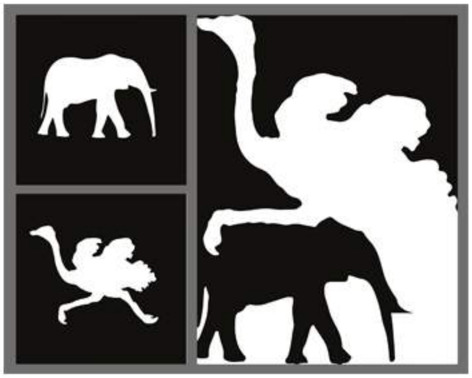


Figure 1.27: Examples of nested images [61].

In chapters of results the advances related to the use of emerging images for the design of HIPs will be presented.

Embedded images in this thesis are used to develop an interactive test with humans. Furthermore, in the following sections we present some concepts related to the Brauer configurations, posets (partially ordered sets), poset representation theory, tiled orders, and topological data analysis. These structures will be used to obtain emerging images that will be used in human interactive proofs.

1.6 Representation of Algebras

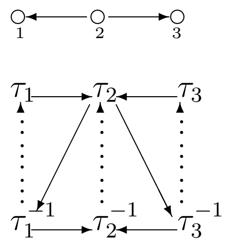
In this section, we describe the main definitions, notations and results regarding classification of algebras which will be used throughout this document [11,45,48].

Definition 1.6.1. A *quiver* $Q = (Q_0, Q_1, s, t)$ is a quadruple consisting of two sets Q_0, whose elements are called *vertices* and Q_1, whose elements are called arrows, s and t are maps $s, t : Q_1 \to Q_0$ such that if α is an arrow then $s(\alpha)$ is called the *source* of α and $t(\alpha)$ is called the *target* of α.

Definition 1.6.2. Let Q be a quiver. A *path* of length $l \geq 1$ in a quiver Q, with source a and target b is a finite sequence $(a \mid \alpha_1, \alpha_2, \ldots, \alpha_l \mid b)$, where $t(\alpha_i) = s(\alpha_{i+1})$ for any $1 \leq i < l$. For each vertex $v \in Q_0$, we define a path of length zero e_v with $s(e_v) = v = t(e_v)$.

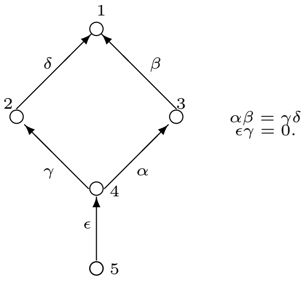
Definition 1.6.3. If Q is a quiver and $\mathbf{k}$ is an algebraically closed field, then the *path algebra* $\mathbf{k}Q$ of Q is the $\mathbf{k}$-algebra whose underlying $\mathbf{k}$-vector space has as a basis the set of all paths of length $l \geq 0$ in Q, and the product of two paths is concatenation if exists or zero otherwise.

Example 1.6.4.



A **k**-algebra A is said to be *basic* if it has a complete set $\{e_1, e_2, \ldots, e_l\}$ of primitive orthogonal idempotents, such that $e_iA \ncong e_jA$ for all $i \neq j$. A *relation* for a quiver Q is a linear combination of paths of length ≥ 2 with same starting point and same ending point, and not all the coefficients being zero.

Example 1.6.5.



Let Q be a finite and connected quiver. The two sided ideal of the path algebra $\mathbf{k}Q$ generated by the arrows of Q is called the *arrow ideal* of $\mathbf{k}Q$ and is denoted by R_Q, R_Q^l is the ideal of $\mathbf{k}Q$ generated as a **k**-vector space, by the set of all paths of length $\geq l$. A two sided ideal $\mathcal{I}$ of the path algebra $\mathbf{k}Q$ is said to be *admissible* if there exists $m \geq 2$, such that $R_Q^m \subseteq \mathcal{I} \subset R_Q^2$.
If $\mathcal{I}$ is an admissible ideal of $\mathbf{k}Q$, the pair $(Q, \mathcal{I})$ is said to be a *bound quiver.* The quotient algebra $\mathbf{k}Q/\mathcal{I}$ is said to be a *bound quiver algebra.*

Gabriel proved the following result [11, 48]:

Theorem 1.6.6. *Let A be a basic and connected finite dimensional $\boldsymbol{k}$-algebra, with $\boldsymbol{k}$ algebraically closed. There exists a quiver Q_A and an admissible ideal $\mathcal{I}$ of $\boldsymbol{k}Q_A$, such that $A \cong \boldsymbol{k}Q_A/\mathcal{I}$.*

Definition 1.6.7. A **k**-*linear representation* or *representation* M of a quiver Q is a system of the form

$$M = ((M_x, \varphi_\alpha) \mid x \in Q_0, \alpha \in Q_1),$$

where M_x is a **k**-vector space for each $x \in Q_0$ and $\varphi : M_a \to M_b$ is a **k**-linear map associated to each arrow $\alpha : a \to b \in Q_1$.

The main problem in the theory of representation of quivers consists of giving a complete description of indecomposable representations and irreducible morphisms of the category of representations $\mathrm{rep}_{\boldsymbol{k}}\, Q$ *of a given quiver* Q.

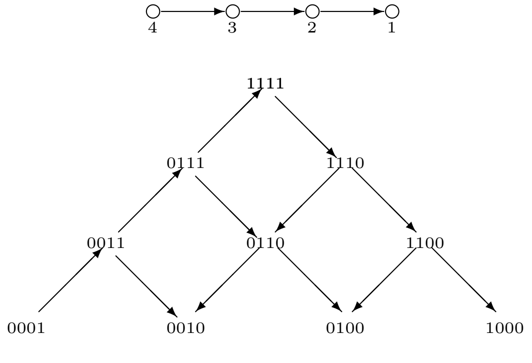
Gabriel also proved the following theorem [11, 48]:

Theorem 1.6.8. *Let* Q *be a finite, connected, and acyclic quiver;* $\boldsymbol{k}$ *be an algebraically closed field, and* $A = \boldsymbol{k}Q$ *be the path* $\boldsymbol{k}$*-algebra of* Q.

1. *The algebra* A *is representation-finite if and only if the underlying graph* $\overline{Q}$ *of* Q *is one of the Dynkin diagrams* $\mathbb{A}_n$, $\mathbb{D}_n$, $\mathbb{E}_6, \mathbb{E}_7$ *and* $\mathbb{E}_8$ *with* $n \geq 4$.
2. *If* $\overline{Q}$ *is a Dynkin graph, then the mapping* $\boldsymbol{dim} : M \to \boldsymbol{dim}\, M$ *induces a bijection between the set of isomorphisms classes of indecomposable* A*-modules and the set of positive roots of the quadratic form* q_Q *of* Q.
3. *The number of the isomorphism classes of indecomposable* A*-modules equals* $\frac{n(n+1)}{2}$, $n^2 - n$, 36, 63 *and* 120, *if* $\overline{Q}$ *is the Dynkin graph* $\mathbb{A}_n$, $\mathbb{D}_n$, $\mathbb{E}_6, \mathbb{E}_7$ *and* $\mathbb{E}_8$ *with* $n \geq 4$.

Definition 1.6.9. The Auslander-Reiten quiver Γ_Q of a quiver Q is a translation quiver with indecomposable representations of the algebra $\mathbf{k}Q$ as vertices and irreducible morphisms as arrows.

Below we show the Auslander-Reiten quiver of the algebra $A = \mathbf{k}(\overrightarrow{\mathbb{A}_4})$, obtained by associating a linear orientation to the Dynkin diagram $\mathbb{A}_4$.



1.6.1 The Drozd's Theorem

Definition 1.6.10. Let $\mathcal{C}$ be a category of (finitely generated)-modules over a $\mathbf{k}$-algebra, where $\mathbf{k}$ is an infinite field, we say that $\mathcal{C}$ has *tame* type if

$\mathcal{C} = \bigcup_n \mathcal{C}_n$ and for every n, the indecomposable modules in $\mathcal{C}_n$ form a one parameter family with maybe finitely many exceptions.

We say that $\mathcal{C}$ has *wild* type if $\mathcal{C}$ contains n-parameter families of indecomposable for arbitrary large n.

Theorem 1.6.11. *Each finite-dimensional algebra A is either wild or tame or finitely represented.*

Definition 1.6.12. Let $\Delta = (\Delta_0, \Delta_1)$ be a graph then the *adjacency matrix* is the $n \times n$-integer matrix A_Δ:

$$(A_\Delta)_{i,j} = \begin{cases} \text{the number of edges between i and j,} & \text{if } i \neq j, \\ \text{two times the number of loops at } i, & \text{if } i = j. \end{cases}$$

Definition 1.6.13. If Λ_Δ is the set of characteristic values of A_Δ then $\max\limits_{\lambda \in \Lambda_\Delta} |\lambda|$ is the *spectral radius* of Δ, denoted $\rho(Q)$.

Theorem 1.6.14. *The largest eigenvalue λ_1 and the maximum vertex degree δ_Δ of a graph Δ are related by the inequality $\sqrt{\delta_\Delta} \leq \lambda_1 \leq \delta_\Delta$.*

Theorem 1.6.15.

1. *A finite connected graph Δ is a laced Dynkin diagram if and only if $\rho(\Delta) < 2$.*

2. *A finite connected graph Δ is an extended Dynkin diagram if and only if $\rho(\Delta) = 2$.*

The following result was proved by M.I. Platzeck and E.Fernández [45].

Theorem 1.6.16. *If Q is a connected quiver without oriented cycles then:*

1. *$\boldsymbol{k}Q$ is of finite representation type if and only if $\rho(Q) < 2$,*

2. *$\boldsymbol{k}Q$ is of tame representation type if and only if $\rho(Q) = 2$,*

3. *$\boldsymbol{k}Q$ is of wild representation type if and only if $\rho(Q) > 2$.*

1.6.2 Brauer Configuration Algebras

Brauer configuration algebras were introduced by Green and Schroll as a way of dealing with the research of algebras of wild representation type [54]. They are associated to a suitable configuration which can be seen as a generalization of a Brauer graph, its definition goes as follows:

A *Brauer configuration* is a tuple $\Gamma = (\Gamma_0, \Gamma_1, \mu, \mathfrak{o})$ where Γ_0 is a set of vertices, Γ_1 is a set of *polygons*, $\mu : \Gamma_0 \to \mathbb{N}$ is a *multiplicity function* and $\mathfrak{o}$ is an orientation, such that the following conditions hold:

$(C1)$ Every vertex in Γ_0 is a vertex in at least one polygon in Γ_1.

$(C2)$ Every polygon has at least two vertices.

$(C3)$ Every polygon in Γ_1 has at least one vertex α such that $\mu(\alpha)val(\alpha) > 1$.

In this case, every polygon is a multiset consisting of vertices, $\mathrm{occ}(\alpha, V)$ denotes the frequency of the vertex α in the polygon V and the *valency* $val(\alpha)$ of the vertex α is defined in such a way that:

$$val(\alpha) = \sum_{W \in \Gamma_1} \mathrm{occ}(\alpha, V).$$

The *cyclic ordering* at vertex α is obtained by linearly ordering the list (i.e., $V_{i_1} < \cdots < V_{i_t}$ and by adding $V_{i_t} < V_{i_1}$). Such a list is said to be the *successor sequence* at α.

A vertex $\alpha \in \Gamma_0$ is said to be *truncated* if $val(\alpha)\mu(\alpha) = 1$, that is, α occurs exactly once in exactly one $V \in \Gamma_1$ and $\mu(\alpha) = 1$. A vertex is *non-truncated* if it is not truncated.

Remark 1.6.17. Often, the notation $V^{(n)}$ will be used to represent a successor sequence of a vertex α if $\mathrm{occ}(\alpha, \mathrm{V}) = n$.

Let $\Gamma = (\Gamma_0, \Gamma_1, \mathfrak{o}, \mu)$ be a Brauer configuration then Γ is said to be *disconnected* if there are two Brauer configurations $\Gamma' = (\Gamma'_0, \Gamma'_1, \mathfrak{o}', \mu')$ and $\Gamma'' = (\Gamma''_0, \Gamma''_1, \mathfrak{O}'', \mu'')$ such that:

1. $\{\Gamma'_0, \Gamma''_0\}$ is a partition of Γ_0,
2. for every polygon $V \in \Gamma_1$, the vertices of V are either all in Γ'_0 or are all in Γ''_0,

3. $\{\Gamma_1', \Gamma_1''\}$ constitutes a partition of Γ_1,

4. μ' (resp, μ'') is a restriction of μ to Γ_0' (resp, Γ_0'') and

5. The orientations $\mathfrak{o}'$ and $\mathfrak{o}''$ are induced by $\mathfrak{o}$.

In this case we write $\Gamma = \Gamma' \cup \Gamma''$, otherwise Γ is said to be connected. Note that, any Brauer configuration is uniquely written as a union of connected Brauer configurations.

The Quiver of a Brauer Configuration Algebra

The quiver Q_Γ of a Brauer configuration algebra is defined in such a way that the vertex set $\{v_1, v_2, \ldots, v_m\}$ of Q_Γ is in correspondence with the set of polygons $\{V_1, V_2, \ldots, V_m\}$ in Γ_1, noting that there is one vertex in Q_Γ for every polygon in Γ_1. Arrows in Q_Γ are defined by the successor sequences.

For each non-truncated vertex $\alpha \in \Gamma_0$ and each successor V' of V at α, there is an arrow from v to v' in Q_Γ where v and v' are the vertices in Q_Γ associated to the polygons V and V' in Γ_1, respectively.

For example, consider a configuration $\Gamma = (\Gamma_0, \Gamma_1, \mu, \mathfrak{o})$ such that

1. $\Gamma_0 = \{1, 2, 3, 4\}$,

2. $\Gamma_1 = \{U = \{1, 1, 2, 3, 3, 4\}, V = \{1, 2, 3, 4, 4, 4\}\}$,

3. At vertex 1, it holds that; $U < U < V, \quad val(1) = 3$,

4. At vertex 2, it holds that; $U < V, \quad val(2) = 2$,

5. At vertex 3, it holds that; $U < U < V, \quad val(3) = 3$

6. At vertex 4, it holds that; $U < V < V < V, \quad val(4) = 4$,

7. $\mu(\alpha) = 1$ for any vertex α.

Figure 1.6.2 shows the quiver Q_Γ associated to this configuration and corresponding indecomposable projective modules P_U and P_V.

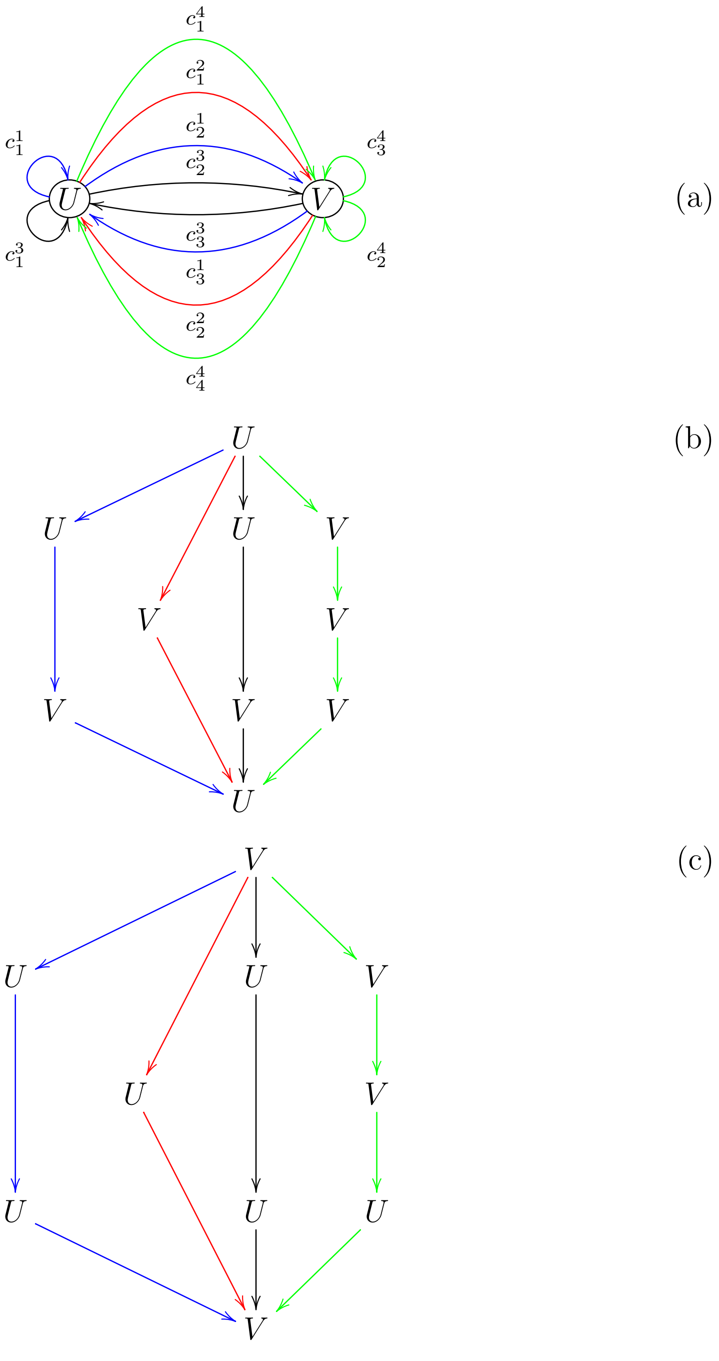


Figure 1.28: Q_Γ and corresponding indecomposable projective modules

The Ideal of Relations and Definition of a Brauer Configuration Algebra

Fix a polygon $V \in \Gamma_1$ and suppose that $\operatorname{occ}(\alpha, V) = t \geq 1$ then there are t indices $i_1, \dots, i_t$ such that $V = V_{i_j}$. Then the *special α-cycles* at v are the cycles $C_{i_1}, C_{i_2}, \dots, C_{i_t}$ where v is the vertex in the quiver of Q_Γ associated to the polygon V. If α occurs only once in V and $\mu(\alpha) = 1$ then there is only one special α-cycle at v.

The ideal of relations I_Γ of the Brauer configuration algebra associated to the Brauer configuration Γ is generated by three types of relations:

1. **Relations of type I**. For each polygon $V = \{\alpha_1, \dots, \alpha_m\} \in \Gamma_1$ and each pair of non-truncated vertices α_i and α_j in V, the set of relations ρ_Γ contains all relations of the form $C^{\mu(\alpha_i)} - C'^{\mu(\alpha_j)}$ where C is a special α_i-cycle and C' is a special α_j-cycle.

2. **Relations of type II**. Relations of type II are all paths of the form $C^{\mu(\alpha)}a$ where C is a special α-cycle and a is the first arrow in C.

3. **Relations of type III**. These relations are quadratic monomial relations of the form ab in $\mathbf{k}Q_\Gamma$ where ab is not a subpath of any special cycle unless $a = b$ and a is a loop associated to a vertex of valency 1 and $\mu(\alpha) > 1$.

Let $\mathbf{k}$ be a field and Γ a Brauer configuration. The *Brauer configuration algebra associated to* Γ is defined to be $\mathbf{k}Q_\Gamma/I_\Gamma$, where Q_Γ is the quiver associated to Γ and I_Γ is the ideal in $\mathbf{k}Q_\Gamma$ generated by the set of relations ρ_Γ of type I, II and III.

The following results give some description of the structure of Brauer configuration algebras [54].

Theorem 1.6.18. *Let Λ be a Brauer configuration algebra with Brauer configuration Γ.*

1. *There is a bijective correspondence between the set of projective indecomposable Λ-modules and the polygons in Γ.*

2. *If P is a projective indecomposable Λ-module corresponding to a polygon V in Γ. Then* $\operatorname{rad} P$ *is a sum of r indecomposable uniserial modules, where r is the number of (non-truncated) vertices of V and where the intersection of any two of the uniserial modules is a simple Λ-module.*

3. *A Brauer configuration algebra is a multiserial algebra.*

4. *The number of summands in the heart of an indecomposable projective Λ-module P such that* $\operatorname{rad}^2 P \neq 0$ *equals the number of non-truncated vertices of the polygons in Γ corresponding to P counting repetitions.*

Proposition 1.6.19. *Let Λ be the Brauer configuration algebra associated to the Brauer configuration Γ. For each $V \in \Gamma_1$ choose a non-truncated vertex α and exactly one special α-cycle C_V at V then*

$\{\overline{p} \mid p\, is\, a\, proper\, prefix\, of\, some\, C^{\mu(\alpha)}\, where\, C\, is\, a\, special\, \alpha{-}cycle\} \bigcup \{\overline{C^{\mu(\alpha)}} \mid V \in \Gamma_1\}$ *is a $\boldsymbol{k}$-basis of Λ.*

Proposition 1.6.20. *Let Λ be a Brauer configuration algebra associated to the Brauer configuration Λ and let $\mathcal{C} = \{C_1, \ldots, C_t\}$ be a full set of equivalence class representatives of special cycles. Assume that for $i = 1, \ldots, t$, C_i is a special α_i-cycle where α_i is a non-truncated vertex in Γ. Then*

$$\dim_{\boldsymbol{k}} \Lambda = 2|Q_0| + \sum_{C_i \in \mathcal{C}} |C_i|(n_i|C_i| - 1),$$

where $|Q_0|$ denotes the number of vertices of Q, $|C_i|$ denotes the number of arrows in the α_i-cycle C_i and $n_i = \mu(\alpha_i)$.

1.7 Partially Ordered Sets (Posets)

In this section we introduce some basic definitions and notation regarding posets (see [24, 27, 29, 37] for more detailed definitions) which can be used to describe more ahead nested emerging and multistable images.

Definition 1.7.1. An *ordered set* (or *Partially ordered set* or *poset*) is an ordered pair of the form $(\mathcal{P}, \leq)$ of a set $\mathcal{P}$ and a binary relation $\leq$ contained in $\mathcal{P} \times \mathcal{P}$, called the *order* (or the *partial order*) on $\mathcal{P}$, such that $\leq$ is reflexive, antisymmetric and transitive [37]. The elements of $\mathcal{P}$ are called the *points* of the ordered set. We will write $x < y$ for $x \leq y$ and $x \neq y$, in this case we will say x is *strictly less than* y. An ordered set will be called *finite (infinite)* if and only if the underlying set is finite (infinite). Usually we shall be a little slovenly and say simply $\mathcal{P}$ *is an ordered set*. Where it is necessary to specify the order relation overtly we write $(\mathcal{P}, \leq)$.

Let $\mathcal{P}$ be an ordered set and let $x, y \in \mathcal{P}$ we say x is covered by y if $x < y$ and $x \leq z < y$ implies $z = x$.

Let $\mathcal{P}$ be a finite ordered set. We can represent $\mathcal{P}$ by a configuration of circles (representing the elements of $\mathcal{P}$) and interconnecting lines (indicating the covering relation). The construction is as follows.

(1) To each point $x \in \mathcal{P}$, associate a point $p(x)$ of the Euclidean plane $\mathbb{R}^2$, depicted by a small circle with center at $p(x)$.

(2) For each covering pair $x < y$ in $\mathcal{P}$, take a line segment $l(x, y)$ joining the circle at $p(x)$ to the circle at $p(y)$.

(3) Carry out (1) and (2) in such a way that

   (a) if $x < y$, then $p(x)$ is lower than $p(y)$,

   (b) the circle at $p(z)$ does not intersect the line segment $l(x, y)$ if $z \neq x$ and $z \neq y$.

A configuration satisfying (1)-(3) is called a *Hasse diagram* or *diagram* of $\mathcal{P}$. In the other direction, a diagram may be used to define a finite ordered set; an example is given below.

Example 1.7.2.

For the ordered set $\mathcal{P} = \{a, b, c, d, e, f\}$, in which $a < b < c < d < e$, and $f < c$. (See Figure 1.29)

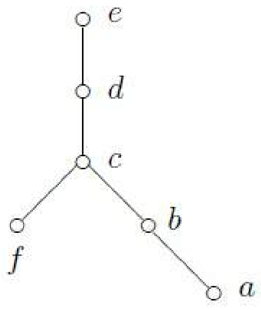


Figure 1.29: Example of Poset.

We have only defined diagrams for finite ordered sets. It is not possible to represent the whole of an infinite ordered set by a diagram, but if its structure is sufficiently regular it can often be suggested diagrammatically. Of course, the same ordered set may have different diagrams. Diagram-drawing is as much an art as a science, and, as we will see, good diagrams can be a real asset to understanding and to theorem-proving.

An ordered set C is called a *chain* (or a *totally ordered set* or a *linearly ordered set*) if and only if for all $p, q \in C$ we have $p \leq q$ or $q \leq p$ (i.e., p and q are comparable). If any two of points of C are incomparable then C is said to be an antichain [10, 37, 80].

1.7.1 Equipped Posets

In this section, we define equipped posets and the category of representations of this kind of posets [24, 74].

A poset $(\mathcal{P}, \leq)$ is called *equipped* if all the order relations between its points $x \leq y$ are separated into strong (denoted $x \trianglelefteq y$) and weak (denoted $x \preceq y$) in such a way that

$$x \leq y \trianglelefteq z \quad \text{or} \quad x \trianglelefteq y \leq z \quad \text{implies} \quad x \trianglelefteq z. \tag{1.4}$$

i.e., a composition of a strong relation with any other relation is strong.

In general relations $\trianglelefteq$ and $\preceq$ are not order relations. These relations are antisymmetric but not reflexive. In particular $\preceq$ is not reflexive (meanwhile $\trianglelefteq$ is transitive) [24, 74].

We let $x \leq y$ denote an arbitrary relation in an equipped poset $(\mathcal{P}, \leq)$. The order $\leq$ on an equipped poset $\mathcal{P}$ gives raise to the relations $\prec$ and $\lhd$ of *strict inequality*: $x \prec y$ (respectively, $x \lhd y$) in $\mathcal{P}$ if and only if $x \preceq y$ (respectively, $x \trianglelefteq y$) and $x \neq y$.

A point $x \in \mathcal{P}$ is called *strong* (*weak*) if $x \trianglelefteq x$ (respectively, $x \preceq x$). These points are denoted $\circ$ (respectively, $\otimes$) in diagrams. We also denote $\mathcal{P}^{\circ} \subseteq \mathcal{P}$ (respectively, $\mathcal{P}^{\otimes} \subseteq \mathcal{P}$) the subset of strong points (respectively, weak points) of $\mathcal{P}$. If $\mathcal{P}^{\otimes} = \varnothing$ then the equipment is *trivial* and the poset $\mathcal{P}$ is ordinary.

Remark 1.7.3. Note that if $x \preceq y$ in an equipped poset $(\mathcal{P}, \leq)$ and there exists $t \in \mathcal{P}$ such that $x \leq t \leq y$ then $x, y \in \mathcal{P}^{\otimes}$, $x \preceq t$ and $t \preceq y$. Otherwise, if $x \trianglelefteq t$ or $t \trianglelefteq y$ then by definition it is obtained the contradiction $x \trianglelefteq y$.

If $\mathcal{P}$ is an equipped poset and $a \in \mathcal{P}$ then the subsets of $\mathcal{P}$ denoted $a^{\vee}$, $a_{\wedge}$, $a^{\triangledown}$, $a_{\vartriangle}$, $a^{\blacktriangledown}$, $a_{\blacktriangle}$, $a^{\curlyvee}$ and $a_{\curlywedge}$ are defined in such a way that:

$$\begin{aligned}
&a^{\vee} = \{x \in \mathcal{P} \mid a \leq x\}\,, \quad a_{\wedge} = \{x \in \mathcal{P} \mid x \leq a\}\,,\\
&a^{\triangledown} = \{x \in \mathcal{P} \mid a \trianglelefteq x\}\,, \quad a_{\vartriangle} = \{x \in \mathcal{P} \mid x \trianglelefteq a\}\,,\\
&a^{\blacktriangledown} = a^{\vee} \setminus a, \quad a_{\blacktriangle} = a_{\wedge} \setminus a,\\
&a^{\curlyvee} = \{x \in \mathcal{P} \mid a \preceq x\}\,, \quad a_{\curlywedge} = \{x \in \mathcal{P} \mid x \preceq a\}\,.
\end{aligned}$$

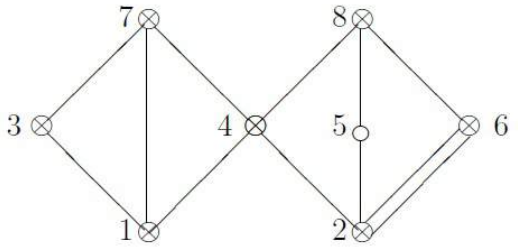
Subset $a^{\vee}$ ($a_{\wedge}$) is called the *ordinary upper (lower) cone*, associated to the point $a \in \mathcal{P}$ and subset $a^{\triangledown}$ ($a_{\vartriangle}$) is called the *strong upper (lower) cone* associated to the point $a \in \mathcal{P}$. Whereas subsets $a^{\blacktriangledown}$ and $a_{\blacktriangle}$ are called *truncated cones* (upper and lower) associated to the point $a \in \mathcal{P}$.

In general, subsets $a^{\curlyvee}$ and $a_{\curlywedge}$ are not cones. Note that, if $x \in \mathcal{P}^{\circ}$ then $x^{\curlyvee} = x_{\curlywedge} = \varnothing$.

The diagram of an equipped poset $(\mathcal{P}, \leq)$ may be obtained via its Hasse diagram (with strong ($\circ$) and weak points ($\otimes$)). In this case, a new line is added to the line connecting two points $x, y \in \mathcal{P}$ with $x \lhd y$ if and only if such relation cannot be deduced of any other relations in $\mathcal{P}$.

Example 1.7.4.

In Figure 1.30, an example of this kind of diagrams is showed.



$1^{\curlyvee} = \{1,3,4,8\}$	$1^{\triangledown} = \{7\}$
$2^{\curlyvee} = \{2,4,7\}$	$2^{\triangledown} = \{5,6,8\}$
$3^{\curlyvee} = \{3,7\}$	$3^{\triangledown} = \varnothing$
$4^{\curlyvee} = \{4,7,8\}$	$4^{\triangledown} = \varnothing$
$5^{\curlyvee} = \varnothing$	$5^{\triangledown} = \{5,8\}$
$6^{\curlyvee} = \{6,8\}$	$6^{\triangledown} = \varnothing$
$7^{\curlyvee} = \{7\}$	$7^{\triangledown} = \varnothing$
$8^{\curlyvee} = \{8\}$	$8^{\triangledown} = \varnothing$

Figure 1.30: Example of equipped Poset [24]

For an equipped poset $(\mathcal{P}, \leq)$ and $A \subset \mathcal{P}$, we define the subsets, $A^{\triangledown}$, $A^{\curlyvee}$ and $A^{\vee}$ in such a way that

$$A^{\triangledown} = \bigcup_{a \in A} a^{\triangledown}, \quad A^{\curlyvee} = \bigcup_{a \in A} a^{\curlyvee}, \quad A^{\vee} = \bigcup_{a \in A} a^{\vee}$$

Subsets $A_{\vartriangle}$, $A_{\curlywedge}$ and $A_{\wedge}$ are defined in the same way.

If $\mathcal{P}$ is an equipped poset then a *chain* $C = \{c_i \in \mathcal{P} \mid 1 \leq i \leq n,\ c_{i-1} < c_i \text{ if } i \geq 2\} \subseteq \mathcal{P}$ is a *weak chain* if and only if $c_{i-1} \prec c_i$ for each $i \geq 2$. If $c_1 \prec c_n$ then we say that C is a *completely weak chain*. Moreover, a subset $X \subset \mathcal{P}$ is *completely weak* if $X = X^{\otimes}$ and weak relations are the only relations between points of X. Often, we let $\{c_1 \prec c_2 \prec \cdots \prec c_n\}$ denote a weak chain which consists of points $c_1, c_2 \ldots, c_n$. An ordinary chain C is denoted in the same way (by using the corresponding symbol $<$).

For an equipped poset $\mathcal{P}$ and $A, B \subset \mathcal{P}$ we write $A < B$ if $a < b$ for each $a \in A$ and $b \in B$. Notations $A \prec B$ and $A \lhd B$ are assumed in the same way.

1.8 Poset Representation Theory

The theory of representation of Posets (partially ordered sets) was developed at the 1970's in Kiev by Nazarova, Roiter and their students. The main goal of their investigations was to obtain a complete classification of the indecomposable objects of the additive category rep $\mathcal{P}$ for a given poset $\mathcal{P}$ [10, 48, 55, 70, 84].

In this case, a representation U of a given poset $(\mathcal{P}, \leq)$ over a commutative ring k is a system of the form:

$$U = (U_0, U_x \mid x \in \mathcal{P}) \tag{1.5}$$

where U_0 is a finite dimensional k-vector space and for each $x \in \mathcal{P}$, U_x is a subspace of U_0 such that $U_x \subseteq U_y$ provided $x \leq y$ [23].

The direct sum of U and V is defined by formula

$$U \oplus V = (U_0 \oplus V_0; U_x \oplus V_x \mid x \in \mathcal{P}) \tag{1.6}$$

A representation $U \in$ *rep* $\mathcal{P}$ is said to be indecomposable if $U \neq 0$ and is not a direct sum of two non-zero representations. Actually, the propose of studying the theory of representation of posets is to give a complete description of the indecomposable objects of the Krull-Schmidt category *rep* $\mathcal{P}$.

Attached to each representation U there exists its matrix presentation $M = M_U$ choosing some basis B_0 in U_0 and for each $x \in \mathcal{P}$, some system B_x of linearly independent generators of U_x module rad $U_x = \overline{U}_x = \sum\limits_{y \preceq x} U_y$. Then

$$M = \begin{array}{|c|c|c|c|} \hline M_{x_1} & M_{x_2} & \dots & M_{x_n} \\ \hline \end{array}$$

with entries in k, partitioned horizontally into $n = |\mathcal{P}|$ (vertical) blocks (also called strips) [55, 84].

Two representations A and B of a poset $(\mathcal{P}, \preceq)$ are isomorphic if and only if their matrix presentations can be turned into each other with help of the following admissible transformations:

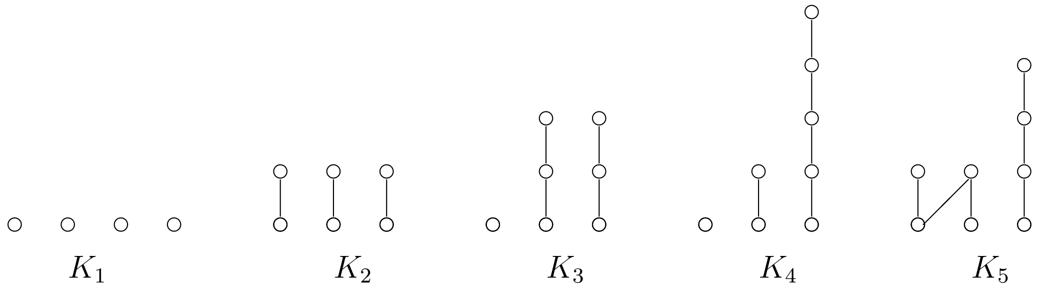
1. elementary transformations of rows of the whole matrix A;

2. elementary transformations of columns within each vertical strip;

3. additions of columns of a strip A_i to columns of a strip A_j if $i \preceq j$ in $\mathcal{P}$.

The following result regarding classification of posets of finite representation type was obtained by Kleiner [55, 59].

Theorem 1.8.1. *A finite poset $\mathcal{P}$ is of finite representation type if and only if $\mathcal{P}$ does not contain as subposet one of the following list*



Often, posets $K_1 \ldots, K_5$ are called the *Kleiner's critical.*

1.9 Tiled Orders

In this section, semimaximal rings also called tiled orders are introduced, this type of rings were introduced and classified by Zavadskij and Kirichenko [27, 100, 101]. In this work, semimaximal rings are used to build emerging images.

A field T is said to be of *discrete norm* or *discrete valuation* if it is endowed with a surjective map [100, 101]

$$\nu : T \to \mathbb{Z} \cup \{\infty\},$$

which satisfies the following conditions :

(a) $\nu(x) = \infty$ if and only if $x = 0$,

(b) $\nu(xy) = \nu(x) + \nu(y)$,

(c) $\nu(x + y) \geq \min\{\nu(x), \nu(y)\}$.

We let $\mathbb{O}$ denote, the *normalization ring* of the field T, such that

$$\mathbb{O} = \{x \in T \mid \nu(x) \geq 0\}.$$

An element $\pi \in \mathbb{O}$ such that $\nu(\pi) = 1$ is a *prime element* of $\mathbb{O}$. For each $x \in \mathbb{O}$, we have that $x \in \mathbb{O}$ if and only if $x = \varepsilon\pi^m$, for some $m \geq 0$ and $\varepsilon \in \mathbb{O}^*$. Moreover, $x \in T$ if and only if $x = \varepsilon\pi^m$ for some $m \in \mathbb{Z}$ and $\varepsilon \in \mathbb{O}^*$.

Ring $\mathbb{O}$ is such that $\mathbb{O} \supset \pi\mathbb{O}$, where $\pi\mathbb{O}$ is the unique maximal ideal, therefore ideals of $\mathbb{O}$ generate a chain of the form

$$\mathbb{O} \supset \pi\mathbb{O} \supset \pi^2\mathbb{O} \supset \cdots \supset \pi^m\mathbb{O} \supset \ldots$$

A *tiled order* or *semimaximal ring* Λ is a subring of the matrix algebra $T^{n\times n}$ with the form

$$\Lambda = \sum_{i,j=1}^{n} e_{ij}\pi^{\lambda_{ij}}\mathbb{O} = \begin{bmatrix} \mathbb{O} & \pi^{\lambda_{12}}\mathbb{O} & \ldots & \pi^{\lambda_{1n}}\mathbb{O} \\ \pi^{\lambda_{21}}\mathbb{O} & \mathbb{O} & \ldots & \pi^{\lambda_{2n}}\mathbb{O} \\ \vdots & \vdots & \vdots & \vdots \\ \pi^{\lambda_{n1}}\mathbb{O} & \pi^{\lambda_{n2}}\mathbb{O} & \ldots & \mathbb{O} \end{bmatrix}.$$

Λ consists of all matrices whose entries ij belong to $\pi^{\lambda_{ij}}\mathbb{O}$, in this case the $e_{ij} \in T^{n\times n}$ are unit matrices such that $e_{ij}e_{kl} = \delta_{jk}e_{il}$ ($\delta_{jk} = 1$, if $j = k$, $\delta_{jk} = 0$ otherwise). Numbers λ_{ij} are integers which satisfy the following conditions:

(1) $\lambda_{ii} = 0$, for each i,

(2) $\lambda_{ij} + \lambda_{jk} \geq \lambda_{ik}$ for all i, j, k.

An order Λ is said to be *Morita reduced* or *reduced* if it satisfies the additional condition:

(3) $\lambda_{ij} + \lambda_{ji} > 0$, for each $i \neq j$. In such a case, projective modules are pairwise non-isomorphic, that is, in the decomposition of $\Lambda = P_1 \oplus P_2 \oplus \cdots \oplus P_n$ via projective modules (i.e., the rows of Λ) all indecomposable projective summands are pairwise not isomorphic, i.e., $P_i \not\simeq P_j$ if $i \neq j$.

The main problem for tiled orders consists of describing all finitely generated torsionless Λ-modules which are called *admissible modules*.

A Λ-admissible right module (not null) is said to be irreducible if it is a submodule of the unique simple module (up to isomorphism). For instance, any indecomposable projective module P_i is irreducible. Thus,

$$P_i = (\pi^{\lambda_{i1}}\mathbb{O}, \pi^{\lambda_{i2}}\mathbb{O}, \dots, \pi^{\lambda_{in}}\mathbb{O}),$$

is a finitely generated irreducible Λ-module without $\mathbb{O}$-torsion.

We denote $\Lambda = (\lambda_{ij})_{i,j=1\dots n}$, note that $\Lambda \subset T^{n\times n} = Q = \Lambda \otimes_{\mathbb{O}} T$, where Q is the rational hull of Λ, Rad $Q = 0$ and Λ has an unique right simple T-module (up to isomorphism) denoted $S_R = (T, T, \dots T) = \sum\limits_{i=1}^{n} e_i T$, $\{e_i \mid 1 \leq i \leq n\}$ is the standard basis such that $e_i e_{jk} = \delta_{ij} e_k$. We assume the notation $S_L = (T, T, \dots, T)^t$ for left modules.

Any irreducible right Λ-module A has the form

$$A = (\pi^{\alpha_1}\mathbb{O}, \pi^{\alpha_2}\mathbb{O}, \dots, \pi^{\alpha_n}\mathbb{O}),$$

where $\alpha_i + \lambda_{ij} \geq \alpha_j$, $\alpha_i \in \mathbb{Z}$, $1 \leq i \leq n$. If A is a left module then we have that $\lambda_{ij} + \alpha_j \geq \alpha_i$.

Henceforth, we denote a right (left) module A in the form $A = (\alpha_1, \alpha_2, \dots, \alpha_n)$ ($(\alpha_1, \alpha_2, \dots, \alpha_n)^t$, respectively).

Note that, $A \simeq A'$ if and only if $\alpha_i = \alpha'_i + k$, for some $k \in \mathbb{Z}$ and any $1 \leq i \leq n$.

Irreducible right modules which are contained in a Q-simple module of a Λ order constitute a lattice denoted $\mathcal{L}_R(\Lambda)$.

We denote $\mathcal{P}(\Lambda) = \mathcal{P}_R(\Lambda)$ the subposet of $\mathcal{L}_R(\Lambda)$ which consists of irreducible projective modules, the projective modules P_i are called *principals* where

$$P_i = (\lambda_{i1}, \lambda_{i2}, \ldots, \lambda_{in}) = P_i^0, \;\; P_i^0 \in \mathcal{P}_R(\Lambda)$$

In this poset there are as many projective modules as infinite chains. In such a case, modules of the form

$$P_i^k = (\lambda_{i1} + k, \ldots, \lambda_{in} + k), \; k \in \mathbb{Z}$$

are projective modules isomorphic to P_i^0. Therefore:

$$\mathcal{P}(\Lambda) = \{P_i^k \mid 1 \leq i \leq n\}$$

where

$$P_i^k \leq P_j^l \text{ if and only if } \begin{cases} k - l \geq \lambda_{ij}, & \mathcal{P}_L(\Lambda), \\ k - l \geq \lambda_{ji}, & \mathcal{P}_R(\Lambda). \end{cases}$$

Thus, the poset $\mathcal{P}(\Lambda)$ is infinite, periodic and the sum of n chains with the form $\{P_i^k \mid 1 \leq i \leq n, k \in \mathbb{Z}\}$, with width $w\left(\mathcal{P}(\Lambda)\right) \leq n$. For instance, the tiled order

$$\Lambda = \begin{bmatrix} 0 & 2 & 2 \\ 2 & 0 & 2 \\ 2 & 2 & 0 \end{bmatrix} = \begin{bmatrix} \mathbb{O} & \pi^2\mathbb{O} & \pi^2\mathbb{O} \\ \pi^2\mathbb{O} & \mathbb{O} & \pi^2\mathbb{O} \\ \pi^2\mathbb{O} & \pi^2\mathbb{O} & \mathbb{O} \end{bmatrix},$$

is associated the poset of projective modules presented in Figure 1.31.
The map $\sigma : \mathcal{P}_R(\Lambda) \to \mathcal{P}_L(\Lambda)$ given by $\sigma(P_i^k) = P_i^{-k}$ is a natural poset antiisomorphism, thus the pair $\{\mathbb{O}, \mathcal{P}(\Lambda)\}$ defines the tiled order Λ up to isomorphism, in the following sense:

$$\Lambda \simeq \Lambda^{'}\text{if and only if } \{\mathbb{O}, \mathcal{P}(\Lambda)\} \simeq \{\mathbb{O}^{'}, \mathcal{P}(\Lambda^{'})\},$$

i.e. $\mathbb{O} \simeq \mathbb{O}^{'}$ and $\mathcal{P}(\Lambda) \simeq \mathcal{P}(\Lambda^{'})$ which means that there exists a poset isomorphism $\varphi : \mathcal{P}_L(\Lambda) \to \mathcal{P}_L(\Lambda^{'})$ such that $A \simeq B$ if and only if $\varphi(A) \simeq \varphi(B)$. It follows that Λ and $\Lambda^{'}$ are Morita-equivalent provided that φ preserves isomorphisms.

The following result characterizes isomorphic orders via matrix problems [55].

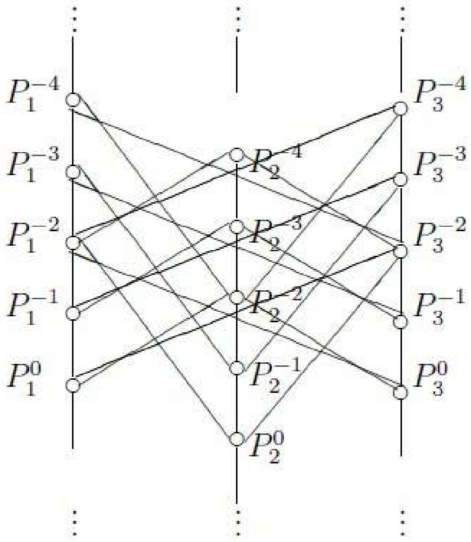


Figure 1.31: Associated poset of irreducible projective modules [27].

Theorem 1.9.1. *Two orders Λ and $\Lambda^{'}$ are isomorphic if the corresponding exponents matrices λ_{ij} and $\lambda_{ij}^{'}$ can be turned into each other with the help of the following admissible t-transformations:*

1. *To add an integer n to each entry of a given row i and simultaneously subtract n to each entry of the column i.*

2. *To transpose simultaneously rows i and j and columns i and j.*

The following is the finite-representation type criterion for tiled orders introduced by Zavadskij and Kirichenko [100, 101].

Theorem 1.9.2. *A tiled order Λ is of finite-representation type if and only if $\mathcal{P}(\Lambda) \not\supset K_1, \ldots, K_5$.*

For $m \geq 1$ a $(0, 1, 2, \ldots, m)$-*tiled order* is a tiled order $\Lambda = (\lambda_{ij})$, $1 \leq i, j \leq n$, where $\lambda_{ij} \in \{0, 1, 2, \ldots, m\}$. In particular, if $\Lambda = (\lambda_{ij})$ is a $(0, m)$-tiled order then Λ has associated a finite poset

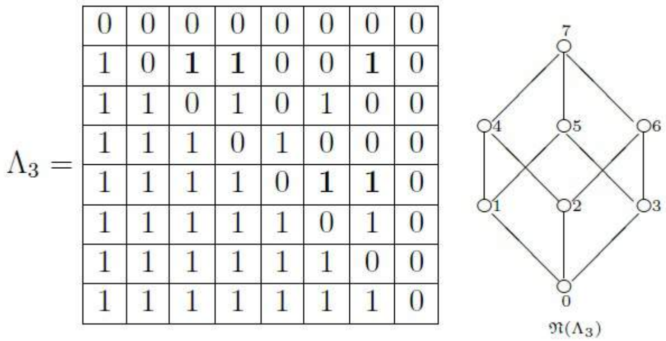
$$(\mathfrak{N}, \leq) = \mathfrak{N}(\Lambda) = (\{1, 2, \ldots, n\}, \leq)$$

where

$$i \leq j \text{ if and only if } \lambda_{ij} = 0.$$

For instance, see Figures 1.32 and 1.33 in which the isomorphic $(0,1)$-orders are presented.

Example 1.9.3.



$$\Lambda_3 = \begin{array}{|c|c|c|c|c|c|c|c|}\hline 0&0&0&0&0&0&0&0\\\hline 1&0&\mathbf{1}&\mathbf{1}&0&0&\mathbf{1}&0\\\hline 1&1&0&1&0&1&0&0\\\hline 1&1&1&0&1&0&0&0\\\hline 1&1&1&1&0&\mathbf{1}&\mathbf{1}&0\\\hline 1&1&1&1&1&0&1&0\\\hline 1&1&1&1&1&1&0&0\\\hline 1&1&1&1&1&1&1&0\\\hline \end{array}$$

Figure 1.32: Tiled (0,1) orders and associated poset [28].

Example 1.9.4.

Actually, infinite families of isomorphic $(0,1)$-tiled orders of these types Λ_n, Γ_n can be constructed.

In such a case, if we denote $\mathcal{S}_{\Lambda_n^{\nearrow}}$ (respectively, $\mathcal{S}_{\Gamma_n^{\swarrow}}$) the size of the support of the upper diagonal (respectively, lower diagonal), then

$$\text{Supp}(\Lambda_n^{\nearrow}) = \sum_{r=0}^{n-2} B_n^r,$$

where $\{B_n^r \mid 0 \leq r \leq n-2\}$ is a set-partition of $\text{Supp}(\Lambda_n^{\nearrow})$ such that:

1. for each $n \geq 0$, $\mid B_n^r \mid$ is a sum of consecutive positive integers $\sum_{k=0}^{T_n^r} g_n^r - k$;

$$\Gamma_3 = \begin{array}{|c|c|c|c|c|c|c|c|}\hline 0&1&1&1&1&1&1&1\\ \hline 0&\mathbf{0}&1&1&1&1&1&1\\ \hline 0&\mathbf{1}&\mathbf{0}&1&1&1&1&1\\ \hline 0&0&0&0&1&1&1&1\\ \hline 0&1&1&1&0&1&1&1\\ \hline 0&\mathbf{0}&1&1&0&\mathbf{0}&1&1\\ \hline 0&\mathbf{1}&\mathbf{0}&1&0&\mathbf{1}&\mathbf{0}&1\\ \hline 0&0&0&0&0&0&0&0\\ \hline\end{array}$$

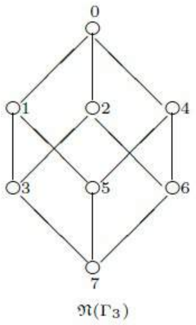


Figure 1.33: Tiled (0,1) orders and associated poset Λ_3 [28].

2.

$$g_n^r = \begin{cases} 2^{n-1} - 1, & \text{if } r = 0, \\ h_n^r = \sum\limits_{s=1}^{n-r-1} [c(s, n-s) - c(s, n-s-r-1)], & \text{if } r \geq 1 \\ t_{n-1} = \frac{(n-1)n}{2}, & \text{if } r = n-1; \end{cases}$$

3.

$$T_n^h = \begin{cases} c(h+1, n), & \text{if } 0 \leq h \leq n-3, \\ n-1, & \text{if } h = n-2, \end{cases}$$

where $c(k, n) = \binom{n+k}{k}$.

Remark 1.9.5. For each r and n fixed, B_n^r consists of subsets ρ_k^r with $\mid \rho_r^k \mid = g_n^r - k$, which defines uniquely a projective module $P_k^0(r) \subset \Lambda_n$.

We recall that A.M. Cañadas et al. used posets Λ and Γ in order to find a formula for the number of two-point antichains in some special posets [28].

1.10 Homological Persistence

Weinberger in [94] made the following statement regarding some open problems in Artificial Intelligence; “Consider the art of Seurat or a piece of old newsprint. The eye, or the brain, performs the marvelous task of taking the sense data of individual points and assembling them into a coherent image of a continuum-it infers the continuous from the discrete" [94]; where the importance of passing since discreet to continuous is presented. See Figure 1.34. This problem is also of our interest. In this section we study some elements of persistent homology.

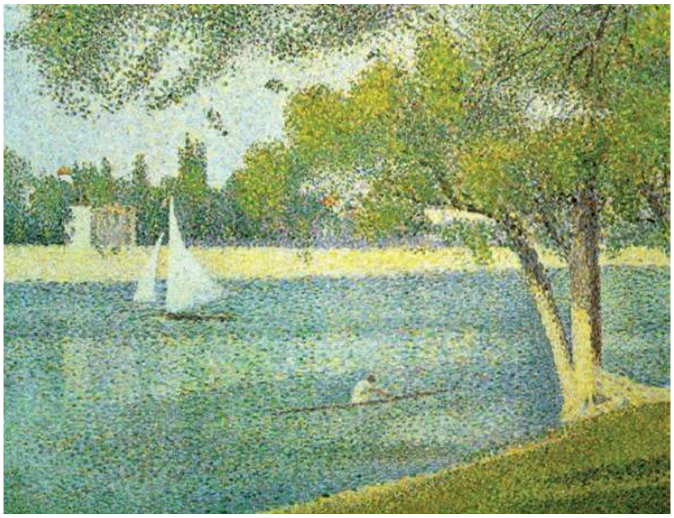


Figure 1.34: Landscape: passing since discreet to continuous [94].

The topological data analysis (TDA) is a field that blend computer science, algebraic topology and statistics. This field is founded on the assumption that scientific dates sets carry information in their internal structure and that sometimes this structure is topological.

The modern theory of persistence is built on three pillars:

1. The persistence diagram and an algorithm for computing it, were introduced by Edelsbrunner, Letscher and Zomorodian. This gives a compact representation of the size function and an effective way to compute it [40–42].

2. Zomorodian and Carlsson defined abstracting persistence modules, indexed by the natural numbers and viewed as graded modules over a polynomial ring [105].

3. Cohen-Steiner, Edelsbrunner and Harer formulated and proved the stability theorem, which guarantees that the persistence diagram is robust to changes in the input data. Robustness is measured in terms of a bottleneck distance between persistence diagrams [36].

The question ¿What is the geometric information that can be gleaned from a data cloud? is studied by Topological Data Analysis.

The principal themes of the work of Carlsson, de Silva, Edelsbrunner, Harer, Zomorodian, and others [30, 42, 104, 105] are the following:

- It is beneficial to replace a set of data points with a family of simplicial complexes, indexed by a proximity parameter. This converts the data set into global topological objects.

- It is beneficial to view these topological complexes through the lens of algebraic topology specifically, via a novel theory of persistent homology adapted to parameterized families.

- It is beneficial to encode the persistent homology of a data set in the form of a parameterized version of a Betti number: a barcode.

To compute information about a topological space using a computer, we need a finite representation of the space, due to their structural simplicity, simplicial complexes are currently a popular representation for topological spaces, so we describe their construction and the important concepts.

Below are some concepts as point cloud, simplicial complex, homology, filtration, barcode diagram, among others.

1.10.1 Point Cloud Dataset

A point cloud dataset usually represents a large finite dataset sampled from a geometrical object in a n-dimensional space, possibly with some noise. In general, a point cloud can be sampled in a n-dimensional metric space [52].

Figure 1.35: Example of points cloud.

1.10.2 Simplicial Complexes

Graphs are not adequate to represent the multifaceted higher dimensional relations in data. A better combinatorial gadget for that is the simplicial complex. They can be constructed using simplices, that is, points, line segments, triangles and tetrahedra.

Definition 1.10.1. A simplicial complex K is a set of objects, $V(K)$, called vertices and a set, $(S(K))$, of finite non-empty subsets of $V(K)$, called simplices such that if $\sigma \subseteq V(K)$ forms a simplex, then any non-empty subset of σ does as well (so not just edges, possibly higher dimensional things as well) [42, 105].

Simplicial complexes are one way to define topological spaces combinatorially. More precisely, a simplicial complex K consists of vertices (0-simplices), edges (1-simplices), triangles (2-simplices), tetrahedra (3-simplices), and higher-dimensional k-simplices (containing $k+1$ vertices, see Figure 1.36) such that:

1. if σ is a simplex in K then K contains all lower-dimensional simplices of σ, and

2. the non-empty intersection of any two simplices in K is a simplex in K.

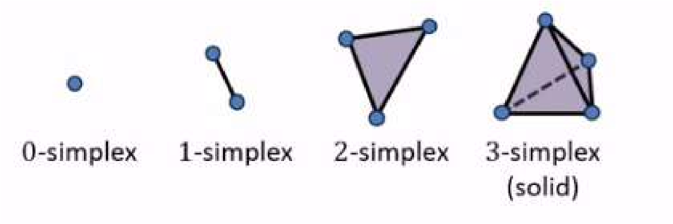


Figure 1.36: Example of simplices.

Example 1.10.2.

$V(K) = \{0, 1, 2, 3, 4\}$

$S(K) = \{\{0\}, \{1\}, \{2\}, \{3\}, \{4\}, \{0, 1\}, \{0, 2\}, \{1, 2\}, \{2, 3\}, \{3, 4\}\}$

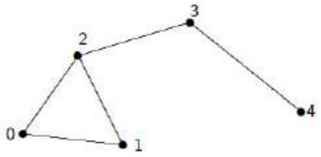


Figure 1.37: Example of simplicial complex.

1.10.3 Homology

By using some algebra, taking formal sums of simplices in a complex, one can get some computable algebraic and numerical invariants of the complex, for instance, the homology groups are defined as $H_i(K) = \text{Ker } \sigma_{i+1}/\text{Img}\sigma_i$, where σ_i and σ_{i+1} are k-linear transformations associated to a simplicial complex K at the ith level. These are typically vector spaces or similar structures, and their dimension tells us the number of holes of different dimension in the space.

1.10.4 Filtration

A filtration of a topological space X is a sequence of subspaces

$$\phi = K_0 \subset K_1 \subset K_2 \cdots \subset K_m = X \tag{1.7}$$

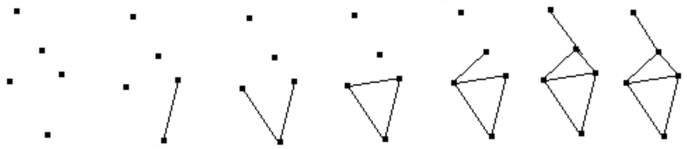


Figure 1.38: Example of filtration.

Definition 1.10.3. Given a simplicial complex K, a *filtration* is a totally ordered set of subcomplexes K_i of K, indexed by the nonnegative integers, such that if $i \leq j$ then $K_i \subseteq K_j$. The total ordering itself is called a *filter*.

1.10.5 Goals of the Homological Persistence

Homological persistence provides a way to relate topological features between different complexes. First we must formalize what types of complexes can be compared.

We assume that our sample of visual data Kcontains finitely many points. Homological persistence is an algebraic method for discerning topological features of dates (components, graph structure, holes) [40]. Homological persistence is also used as an algebraic tool to determine the most persistent topological characteristics of space over time. So that, the purpose of homological persistence is to capture the birth and death times of these components ($0-$dimensional homology classes) and holes ($1-$ dimensional classes) and more generally, higher dimensional homology classes). By birth, we mean a homology class comes into being; by death, we mean it their becomes trivial or becomes identical to some other class born earlier. The persistence or lifetime of a class is the difference between its death and birth times.

Moreover, homological persistence is a mathematical tool from topological data analysis, which performs multi-scale analysis on a set of points and identifies clusters, holes, and voids therein. These latter topological structures complement standard feature representations, making homological persistence an attractive feature extractor for artificial intelligence [102].

1.10.6 Barcode Diagram

Produce barcodes or interval graphs. For each dimension i get a set of closed intervals above an axis parameterized by d.

Definition 1.10.4. A barcode is a collection of half-open subintervals $[b_i, d_j) \subset [0, \infty)$, which describes the homology of the family as it varies over ϵ.

An interval $[b_j, d_j)$ represents a homological feature which is born at time b_j and dies at time d_j .

Long intervals correspond to large holes and thus to genuine features. Small intervals are usually regarded as noise.

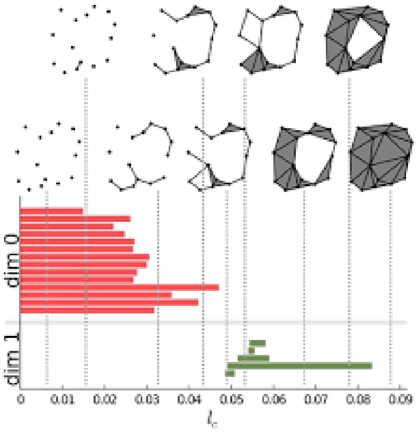


Figure 1.39: Example of Barcode Diagram.

1.10.7 Some Applications of the Theory of Homological Persistence

Some applications of homological persistence are the following, among others: data analysis, coverage of sensor networks, biological networks, medical data analysis, geometric modeling [40].

The first application of Computational Topology methods presented by Edelsbrunner et al. was to the images segmentation. The segmentation problem is to identify regions of interest in an image [42].

Taking images is an efficient way to collect data about the physical world. It can be done fast and in detail. By definition, image processing is the field that concerns itself with the computation aimed at harnessing the information contained in images [85].

In [43] Edelsbrunner claimed that homological persistence is a useful method to quantify and summarize topological information, building a bridge that connects algebraic topology with applications and they showed four technical developments in the overlap between homological persistence and image processing.

One of the first applications of homological persistence to natural language processing is presented by Zhu in [102]. More precisely, Zhu presented an algorithm that identifies holes that can be interpreted as semantic "tie-backs" in a text document, providing a new document structure representation.

1.11 A First Application: Brauer Configuration Algebras to Define a Variable-Length Code

In this section, as a first application of the theory of representation of algebras, we introduce a Brauer configuration whose polygons can be interpreted as codewords of a suitable variable length code (VLC) [22,87].

Definition 1.11.1. Let k and n be positive integers, $k \leq n$, An $[n, k]$ code C, is a k-dimensional subspace of $\mathbb{Z}_2^n$, the vector space of all binary n-tuples.

A generating matrix for an $[n, k]$-code C is a $k \times n$ binary matrix whose rows form a basis for C.

Definition 1.11.2. Let $x, y \in \mathbb{Z}_2^n$, where $x = (x_1, x_2, \ldots, x_n)$ and $y = (y_1, y_2, \ldots, y_n)$. Define the Hamming distance $d(x, y)$ as follows:

$$d(x, y) = |\{i \mid 1 \leq i \leq n, x_i \neq y_i\}|. \tag{1.8}$$

i.e., the number of coordinates in which x and y differ.

Definition 1.11.3. Let C be an $[n, k]$ code. Define the distance of C to be the quantity:

$$d(C) = \min\{d(x, y) \mid x, y \in C, x \neq y\}. \tag{1.9}$$

An $[n, k]$ code with distance d is denoted an $[n, k, d]$ code.

It is worth to note that the parameter n determines the *sufficiency* of the code, k allows to determine how fast the code is, and d measures the capacity of the code to correct possible errors in a communication. The purpose of an *error-correcting code* is to correct random errors that occur in the transmission of (binary) data through a noisy channel.

Let G be a generating matrix for an $[n, k, d]$ code. Suppose x is the binary k-tuple we wish to transmit. Then Alice encodes x as the n-tuple y through the channel. Bob can decode y by using for instance a *nearest neighbor decoding* method which generates at most $(d - 1)/2$ errors.

Definition 1.11.4. If X is an alphabet with cardinality q then a finite sequence $w = x_1 x_2 \ldots x_l$ of code symbols is called a word over X of length $|w| = l$ where $x_i \in X$ for all $i = 1, 2, \ldots, l$.

We let X^+ denote the set of all finite length words over X. The empty word is denoted as λ and $X^* = X^+ \cup \{\lambda\}$. A set of words is called a code.

Definition 1.11.5. Let a code C have s codewords $\{c_1, c_2, \ldots, c_s\}$ and let $l_i = |c_i|$, $i = 1, 2, \ldots, s$. Without loss of generality assume that $l_1 \leq l_2 \leq \ldots, \leq l_s$. Let σ denotes the number of different codewords of positive length in C. If $\sigma = 1$ then C is said to be a fixed-length code, and if $\sigma > 1$ then C is said to be a variable-length code.

An encoded message could either be given as a sequence of source symbols or a sequence of codewords.

In order to retain both the spatial and amplitude information sometimes it is useful to associate to the VLC-codes a rooted tree. In such a case, the root node of the tree represents the start of the message. Each node in the tree is connected to the other s nodes, the s branches connecting these nodes are each labelled with different codewords of C.

In this work, we assume that a quiver Q can be used to define a VLC. In such a case, to each vertex $v \in Q_0$, it is associated a suitable codeword; a message is given by a path P in Q, in such a way that each indecomposable projective module over the algebra kQ can be interpreted as a message generated by the code proposed.

The general setting is defined as follows:

Definition 1.11.6. A variable-length code C_Q associated to a quiver Q is a code whose source symbols are given by the English alphabet $\{a, b, c, \ldots, z\}$ with codewords associated to the vertices of Q.

In this case, C_Q is a matrix whose entries c_{ij} are sequences of source symbols (C_Q has as many rows as $|Q_0|$), and the codewords label vertices of Q. In particular, they are decoded by entries of rows of C_Q. The message space is generated by paths in Q, i.e. the algebra kQ is generated by messages of the code C_Q.

As in the case of autokey in Cryptography goes each sequence $x_1x_2\ldots x_l$ at each entry of C_Q is part of the message making of C_Q a stream code with set of entries $(C_Q)_{i_0,j_0}$ as the initial state.

Definition 1.11.7. Two codes C_Q and C'_Q associated to the same quiver Q are said to be equivalent, denoted $C_Q \sim C'_Q$, provided that they share the same codewords.

The following result regarding codes associated to quivers is easy to prove.

Theorem 1.11.8. *If $C_Q \sim C'_Q$, then C'_Q can be obtained from C_Q by using row and column permutations.*

In the sequel, we define a code associated to the quiver of a Brauer configuration algebra.

Example 1.11.9. Let us to consider the following Brauer configuration:

$$\begin{aligned}
\text{At vertex } a; &\quad K < R < K \\
\text{At vertex } b; &\quad K < Q < R < Q < K \\
\text{At vertex } c; &\quad Q < S < K < R < S < Q \\
\text{At vertex } d; &\quad S < R < N < S \\
\text{At vertex } e; &\quad N < Q < T < N \\
\text{At vertex } f; &\quad N < O < O < N \\
\text{At vertex } g; &\quad O < O < T < O
\end{aligned} \tag{1.10}$$

Then the associated graph is:

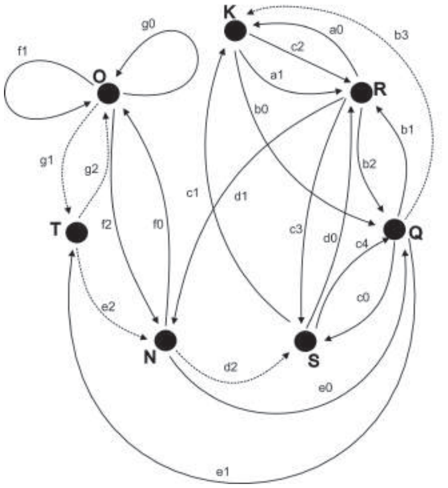


Figure 1.40: A Brauer configuration generating a code C_Q.

The variable-length code C_Q can be defined as follows:

Sequences of Source Symbols				Codewords
bj	hd	ic	ccg	K
icc	jd	fgb	$dcah$	N
gi	dfg	jdc	$jdca$	O
hdg	gic	gbj	hj	Q
bjh	dgi	cgj	jdf	R
$gicc$	gjd	jj	$cahj$	S
jhd	$dgic$	$ccgj$	bjj	T

Note that the following message,

$$g_0 f_0 e_2 e_1 e_0 d_1 d_0 c_3 c_2 c_1 c_0 b_2 b_1 b_0 a_0 a_1 = KRKQRQSKRSRNQTNOO$$

is associated to a sequence of source symbols of the form

bj bjh hd hdg dgi gic gicc ccg cgj gjd jdf fgb gbj bjj jd jdca jdca.

If we apply the assignment $a \to 0$, $b \to 1 \ldots$ (in $\mathbb{Z}_{26}$), then C_Q has the following form

Source Symbols				Codewords
$1,9$	$7,3$	$8,2$	$2,2,6$	10
$8,2,2$	$9,3$	$5,6,1$	$3,2,0,7$	12
$6,8$	$3,5,6$	$9,3,2$	$9,3,2,0$	14
$7,3,6$	$6,8,2$	$6,1,9$	$7,9$	16
$1,9,7$	$3,6,8$	$2,6,9$	$9,3,5$	17
$6,8,2,2$	$6,9,3$	$9,9$	$2,0,7,9$	18
$9,7,3$	$3,6,8,2$	$2,2,6,9$	$1,9,9$	19

and which induces 4 partitions of numbers $10, 12, 14, 16, 17, 18, 19$ (considering 0 as a part).

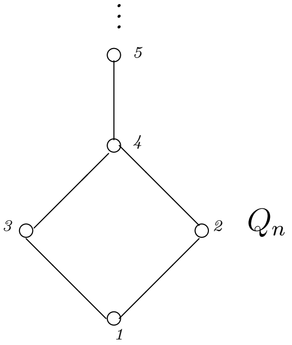
If we consider as initial states sequences at entries $(C_Q)_{1,1}, (C_Q)_{5,1}$, then the message is the integer number

$$c = 197368226935619932079 = KRKQRQSKRSRNQTNOO \tag{1.11}$$

c appears as decimal expansion of the irrational number $\alpha = \sqrt{2 + 2\sqrt{2}}$, which is very useful to classify algebras.

We have the following result as a consequence of Theorem 1.6.16.

Theorem 1.11.10. *If Q_n is a quiver without oriented cycles and such that the underlying diagram $\overline{Q_n}$ has the shape given below, then kQ_n is of wild representation type.*



Proof. If $\rho(Q_n)$ denotes the spectral radius of Q_n then

$$\operatorname*{Lim}_{n\to\infty}\rho(Q_n) = \alpha.$$

Actually, $\rho(Q_{2^n-1})$ is a Cauchy subsequence of $\rho(Q_n)$. □

1.12 Visual Cryptography

Visual Cryptography(VC) proposed by Naor and Shamir [68] is a paradigm for cryptographic schemes that allows the decoding of concealed images without any cryptographic computation. Particulary, in a k-out-of n visual secret sharing scheme (VSS), a secret image is cryptographically encoded into n shares. Each share resembles a random binary pattern. The n shares are then xeroxed onto transparencies respectively and distributed among n participants. The secret images can be visually revealed by stacking together any k or more transparencies of the shares and no cryptographic computation is needed [9]. The following figure 1.41 shows an example of this type of scheme.

1.12.1 Visual Cryptography Schemes

Let $P = \{1, \cdots, n\}$ be a set of elements called *participants* and let 2^P denote the set of all subsets of $\mathcal{P}$. If a participant in not essential, then we can construct a visual cryptography scheme giving him nothing as his share. A nonessential participant does not need to participate actively in the reconstruction of the image.

Visual Secret Sharing (VSSS) is characterized by two parameters: the pixel expansion γ, which is the number of subpixels on each share that each pixel

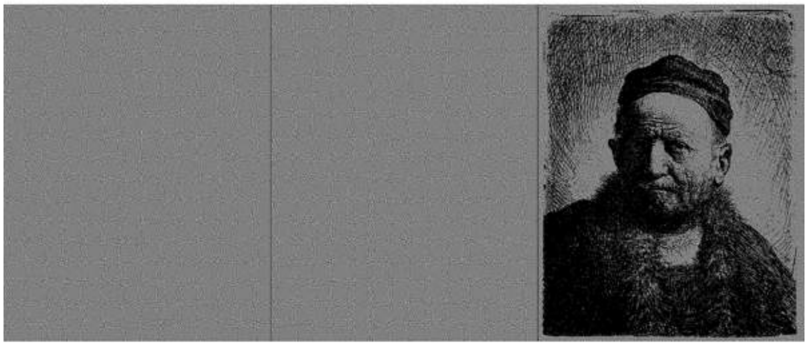


Figure 1.41: Example of a 2-out-2 visual secret sharing scheme.

of the secret image is encoded into, and the contrast α, which is the measurement of the difference of a black pixel and a white pixel in the reconstructed image [19].

VSSS can be applied to access control, so that, they can be used in different types of authentication processes.

The information presented in this chapter will permit to understand how VSSS can be defined via some suitable Brauer configuration algebras. In chapters 2 to 7, we will give more details regarding these processes.

Chapter 2

Emerging and Multistable Images Associated to Brauer Configuration Algebras

In this chapter, we give a Brauer configuration associated to emerging images. This mathematical structure allows us to obtain a good generalization, in order to explain more precisely the relationship between the Brauer configuration algebras and the process of emerging images extraction.

2.1 Brauer Configuration associated to emerging images

If $n = \sum\limits_{i=1}^{k}\alpha_j 10^j$ is a positive integer then we can see n as a Brauer configuration follows the same procedures as described in Section 1.11. In this case, numbers α_j are considered vertices of a Brauer configuration Γ_s and to each of these numbers it is associated a number 0 or 1, $f(\alpha_k = 1)$, $f(\alpha_{k-1} = 0)$ and so on. Where f is the assignment rule, in such a way that $\Gamma_s = \{\Gamma_{s_0}, \Gamma_{s_1}, \mu_s, \mathfrak{o}_s\}$ is defined as follows:

(i) $\Gamma_{s_0} = \{\alpha_1, \alpha_2, \cdots, \alpha_k\}$;

(ii) $\Gamma_{s_1} = \{\mathcal{V}_0 = \{\alpha \mid f(\alpha) = 0\}, \quad \mathcal{V}_1 = \{\alpha \mid f(\alpha) = 1\}\}$;

(iii) $\mu_s(\mathcal{V}_i) = 1$ for all i;

(iv) the orientation is given by the index j for each α_j defining the number n.

A quiver sketch of the Brauer configuration is shown in figure 2.1, where:

c_i^1 corresponds to the number of loops in the polygon $\mathcal{V}_0$;
c_j^1 corresponds to number of transitions of $\mathcal{V}_0$ to $\mathcal{V}_1$;
c_h^2 corresponds to number of loops in the polygon $\mathcal{V}_1$; and
c_k^2 corresponds to number of transitions of $\mathcal{V}_1$ to $\mathcal{V}_0$.

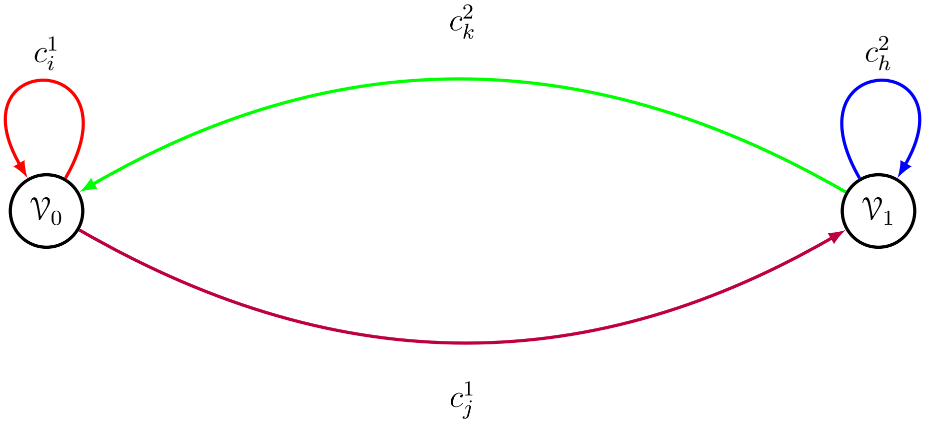


Figure 2.1: Quiver Q_{Γ_s} associated to Γ_s

2.2 Schemes generating from sequencing process

In this section, we present an algorithm of sequencing DSA that allows to take a series of digits and to convert them in a array of 1 or 0, which is associated to Brauer configuration Γ_s. This array constitutes a scheme to generate emerging and multistable images.

Algorithm 2.2.1: DSA: digits sequencing algorithm

1. Read digits of a fixed integer number.
2. Choose the position to select a digit.
3. Define the flag $\mathfrak{f}$ of the searching process. That is, define a digit to be detected throughout the fixed number.
4. Associate either 1 (black) or 0 (white) alternatively to each occurrence of the flag.
5. Define blocks B_i.
6. If B_i is a block of digits between two consecutive flags $\mathfrak{f}_i$ and $\mathfrak{f}_{i+1}$, then each digit $d_i \in B_i$ has the same colour as the assigned to $\mathfrak{f}_i$.
7. Determine transitions or colour changes.
8. Determine successor sequences.

For instance, the number $2^{756.838}(2^{756.839}-1)$ corresponds to the 32th perfect number and has 455.663 digits. Figure 2.2 shows the generated bitmap by using Algorithm 2.2.1. This scheme was obtained computationally by system REIADT (see Experimental Results).

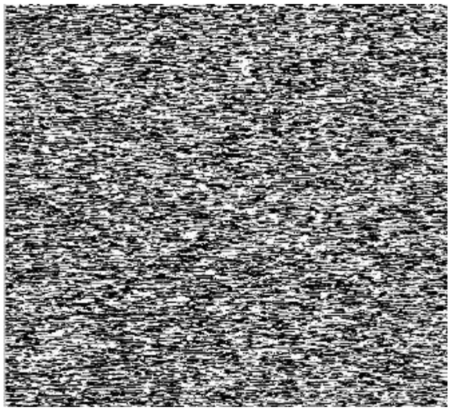


Figure 2.2: Binary master share obtained from a private database encoded by the perfect number $2^{756.838}(2^{756.839}-1)$. It is obtained after a recursive feature level fusion.

2.3 Brauer Configuration associated to digits sequencing

Consider a Brauer configuration $\Gamma_d = \{\Gamma_{d_0}, \Gamma_{d_1}, \mu_d, \mathfrak{o}_d\}$, such that:

(i) $\Gamma_{d_0} = \{x_1, x_2, \cdots, x_n\}$ is a finite integer sequence.

(ii) $\Gamma_{d_1} = \{\mathcal{U} = \{x_1^{f_1}, x_2^{f_2}, \cdots, x_n^{f_n}\}\mathcal{V} = \{x_1^{g_1}, x_2^{g_2}, \cdots, x_n^{g_n}\}\}, f_i, g_i \geq 0$.

(iii) At a vertex x_i, it holds that $\mathcal{U}^{(i_1)} < \mathcal{V}^{(i_2)} < \mathcal{U}^{(i_3)} \cdots$; $val(x_i) = occ(x_i, \mathcal{U}) + occ(x_i, \mathcal{V})$.

(iv) $\mu(\alpha) = 1$ for any vertex.

Remark 2.3.1. By definition the elements in Γ_{d_0} constitute the number $n = x_1 x_2 \cdots x_n$ with $x_1 \neq 0$. Furthermore, the schemes obtained with the sequencing process contain encrypted information obtained by applying recursively a feature level fusion method. And each vertex (block) corresponds to a sequence of integer numbers defining in this way a variable-length code induced by the private database.

2.4 Induced Brauer Configuration

Definition 2.4.1. A Brauer configuration $\Gamma' = E(\Gamma) = (\Gamma'_0, \Gamma'_1, \mu', \mathfrak{o}')$ is said to be induced by the configuration $\Gamma = (\Gamma_0, \Gamma_1, \mu, \mathfrak{o})$ if

$$\left.\begin{array}{l}
\Gamma'_0 = E(\Gamma_0) = \Gamma_0,\\
\Gamma'_1 = E(\Gamma_1) = \{E(\mathcal{U}), E(\mathcal{V})\},\\
\mu' = \mu,\\
\mathfrak{o}' = \mathfrak{o}.
\end{array}\right\} \tag{2.1}$$

where for some multisets $X_0 \subset \mathcal{U}, Y_0 \subset \mathcal{V}$ we have:

$$\left.\begin{array}{l}
E(\mathcal{U}) = (\mathcal{U} \backslash X_0) \cup Y_0;\\
E(\mathcal{V}) = (\mathcal{V} \backslash Y_0) \cup X_0
\end{array}\right\} \tag{2.2}$$

are obtained via an exchange rule.

Definition 2.4.2. Γ and its induced Brauer Configuration $E(\Gamma)$ define the configuration

$$\left.\begin{aligned}
\Gamma\otimes E(\Gamma) &= (\Gamma_0,\Gamma_1^{\otimes},\mu,\mathfrak{o}),\\
\Gamma_1^{\otimes} &= \{\mathcal{U}^{\otimes},\mathcal{V}^{\otimes}\},\\
\mathcal{U}^{\otimes} &= \mathcal{U}\backslash X_0,\\
\mathcal{V}^{\otimes} &= \mathcal{V}\cup X_0.
\end{aligned}\right\}\tag{2.3}$$

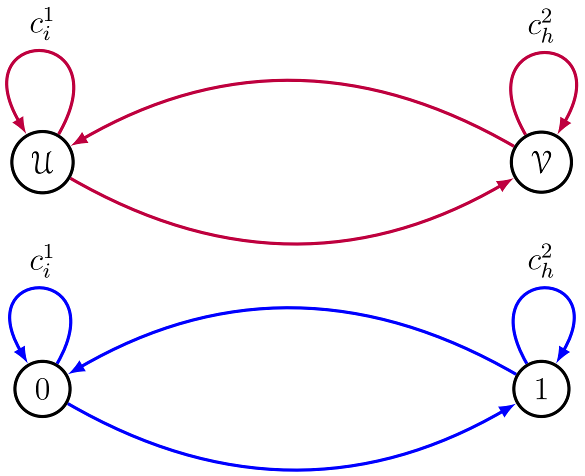


Figure 2.3: Specializations of Brauer Configurations

The specializations of Brauer configurations (see Figure 2.3) allow us to obtain applications in visual sharing schemes, which will be presented in the next section.

2.5 Shares associated to some VSSS

A VSSS can be defined in terms of Brauer configurations as follows.

1. A master share is obtained from a Brauer configuration $\Gamma=(\Gamma_0,\Gamma_1,\mu,\mathcal{O})$, where:

   (i) Γ_0 is a finite sequence of integer numbers (If $n_1=\sum\limits_{j=1}^{k_1}\alpha_j 10^j\in\Gamma_0$ and $n_2=\sum\limits_{i=1}^{k_2}\alpha_i 10^i\in\Gamma_0$ then $\alpha_{k_1}=\alpha_{k_2}$);

(ii) $\Gamma_1 = \{0, 1\} = \{\text{white pixels, black pixels}\}$;

(iii) $\mu(x) = 1$, for any vertex;

(iv) successor sequences are defined according to the place that each vertex has in Γ_0;

(v) slave shares are given by Brauer configurations of the form $E(\Gamma, X_i, Y_i)$, $1 \leq i \leq t$;

(vi) the stacking process is given by the Brauer configuration $\Gamma \otimes E(\Gamma, X_i, Y_i)$, $1 \leq i \leq t$.

Images in these kind of schemes can be obtained after a stacking process (given by the configuration $\otimes$) as in the case of visual sharing schemes. In VSSS images are obtained after such stacking process without any computation. Only the Human Visual System (HVS) should be used to obtain the output-image which generally are emerging images.

For the sake of accuracy, a noise-deletion process is carried out. In order to do that, (see Figure 2.4), it is considered an universal (or master) share and each image is obtained after the following two main steps:

(i) Stacking with a suitable share (slave share).

(ii) Noise deletion. In this step an integer sequence is associated to each share, two different shares have associated different sequences. Such integer sequences allow us to determine which black blocks must be deleted in the corresponding image.

We must bear in mind that, only the dealer (the qualified participant with the master share) can recover totally the integer sequence (or encoding number $A(n)$ associated for a set of vertices in a Brauer configuration) delivered to the participants. $A(n)$ is obtained from a share-integer sequence by concatenation, in such a way that there is a bijection between the vertices of the associated Brauer configuration (black and white blocks) and digits of $A(n)$.

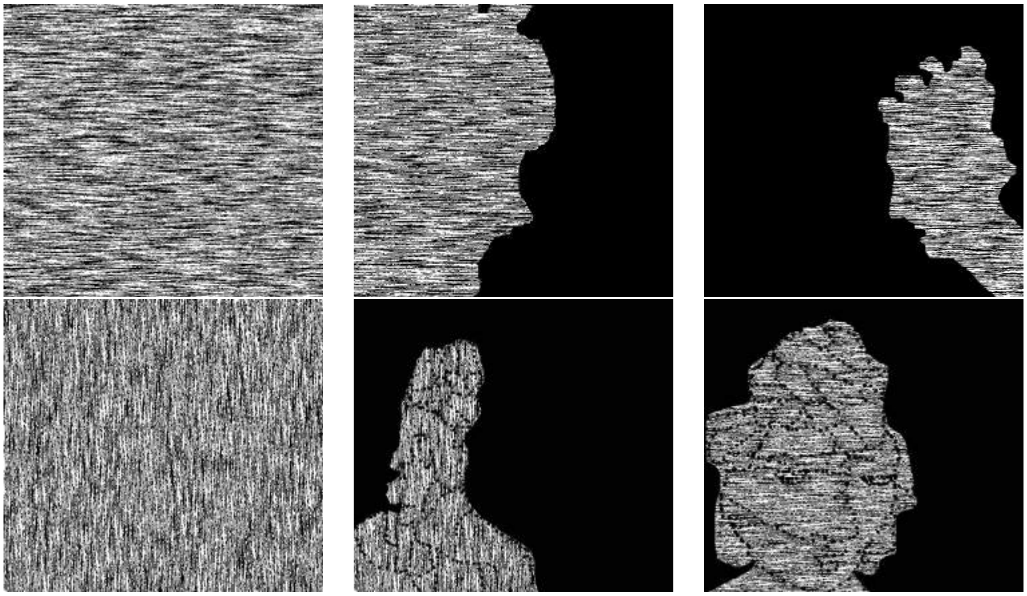


Figure 2.4: Examples: two master binary shares and corresponding slave shares.

2.6 Algebraic Master Share in a Gray-Level and Color VSSS

We next define VSSS for gray-level and color images by using BC's and lattices as follows.

Consider a Brauer configuration $\Gamma = (\Gamma_0, \Gamma_1, \mu, \mathfrak{o})$ such that the following conditions are satisfied:

(i) Γ_0 is a finite lattice with a finite feature-lattice associated Γ_2. Under these circumstances each vertex $x \in \Gamma_0$ is denoted in the form (x, y) with $y \in \Gamma_2$;

(ii) for $1 \leq t \leq k_0$ a polygon

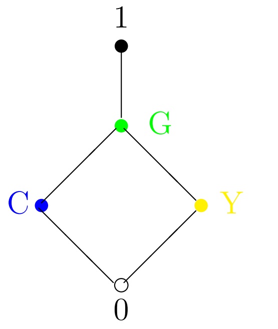
$$U_t = \{(P^t_{i_t}, P^t_{j_t}) \mid P^t_{i_t} \in \Gamma_0, P^t_{j_t} \in \Gamma_2\};$$

(iii) $\mu((P^t_{i_t}, P^t_{j_t})) = n_0$, $n_0 \in \{1, 2, 3, 4\}$ fixed; and

(iv) a successor sequence $U_1 < U_2 \cdots < U_m$ at a vertex $(P^s_{i_s}, P^s_{j_s})$ is interpreted as the joint $\bigvee\limits_{i=1}^{k_0} P^s_{j_i} \in \mathcal{L}_2$.

Example 2.6.1.

The following example assumes that colours constitute a lattice.

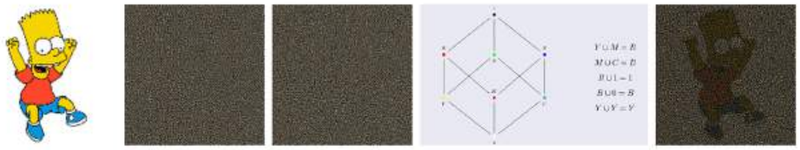


$$\begin{array}{|cc|cc|cc|cc|cc|}
\multicolumn{2}{c}{0} & \multicolumn{2}{c}{Y} & \multicolumn{2}{c}{C} & \multicolumn{2}{c}{G} & \multicolumn{2}{c}{1} \\
\hline
0 & 0 & Y & Y & C & C & G & G & 1 & 1 \\
1 & 0 & G & 1 & Y & 1 & G & 1 & 1 & 1 \\
0 & 1 & 1 & G & 1 & Y & 1 & G & 1 & 1 \\
0 & 0 & Y & Y & C & C & G & G & 1 & 1 \\
\hline
\end{array}$$

$$\underset{F_Y}{\begin{bmatrix} Y \\ 1 \\ 1 \\ Y \end{bmatrix}} \quad \underset{F_C}{\begin{bmatrix} C \\ 1 \\ 1 \\ C \end{bmatrix}} \quad \underset{F_G}{\begin{bmatrix} G \\ 1 \\ 1 \\ G \end{bmatrix}}$$

Example 2.6.2.

The following example shows a 2-out-2 color based VSSS defined by the Brauer configuration Γ.



The following result (given by A.M. Cañadas et al. [29]) defines admissible transformations, which guarantees the existence of equivalent lattice-color matrix representations. Therefore, it guarantees the construction of different types of (k, n) lattice-based VSSS. In this case, admissible transformations correspond to exchange rules in Brauer configurations as defined above.

Theorem 2.6.3. *Let M and M' be two lattice matrix representations of a given lattice $\mathcal{L}$ then M and M' are equivalent if M and M' can be turned one into each other by applying the following transformations:*

(i) *row permutations of the whole matrix;*

(ii) *column permutations within a given vertical stripe;*

(iii) *multiplication of a given column j in the stripe M_x by some scalar $z \in (\lambda_j^x)^{\triangledown}$, where λ_j^x is the maximum of all entries in such a column;*

(iv) *addition of a given jth column in the stripe M_x to the jth column in the stripe M_y with coefficients in $(\delta_j^y)_{\vartriangle}$, where δ_j^y is the minimum of all entries in the column of M_y, provided that $x \leq y$ in $\mathcal{L}$.*

More experimental results regarding Brauer configuration algebras induced by large numbers and its relationships with VSSS are presented in Chapter 7.

Chapter 3

A Mathematical Model for Emerging and Multistable Images

In this chapter, we present an important result of this thesis: a novel mathematical model which uses the Poset Representation Theory for generating emerging and multistable images.

In the following model, each binary image can be seen as a finite poset. The noise in this case is represented by antichains. Therefore, noise in these schemes is deleted by applying successively a completion process.

3.1 Mathematical Model

If M is a k-module, then a model ν of an image repository $\mathcal{D}$ is a map (not necessarily injective) $\nu : \mathcal{D} \to M$ which applies to each element $I \in \mathcal{D}$ an element $\nu(I) = v$. In this case, the element v can be seen as the image I and v is said to be a M-model of I. Henceforth, if a model ν that has been defined on a repository $\mathcal{D}$, then we shall assume that each M-model $\nu(I) \in M$ is again an image giving at least the same visual information provided by I.

As a matter of fact, model $\nu(I)$ is the image obtained from I after applications of some image operations as noise addition, rotation, magnification, translation, reflection, contraction or deformation.

Given an ordered database of binary images:

$$(\mathcal{B}, \preceq) = \{b_1, b_2, \ldots, b_k\},$$

where

$$b_i \preceq b_j \text{ if image } b_i \text{ is embedded in image } b_j,$$

it is defined a X-model of $\mathcal{B}$ with the form:

$$\mathfrak{S} = (X, b_i, \chi, \{X_{b_i^{j_i}}\} \mid 1 \leq j_i \leq t_i, 1 \leq i \leq k), \tag{3.1}$$

where for m, n fixed, $X \subset M_{m \times n}(\mathbb{Z}_2)$ is a finite k-vector space and $\chi \in X$ fixed.

Analogously, (3.1) can be written as:

$$\mathfrak{S} = (\Gamma, b_i, \chi, \{X_{b_i^{j_i}}\} \mid 1 \leq j_i \leq t_i, 1 \leq i \leq k), \tag{3.2}$$

where Γ is a matrix algebra, $\chi \subset M_{m \times n}(\mathbb{Z}_2)$ is a specialization of Λ_Γ.

In fact, entries of matrix χ are pixels induced by $\mathcal{B}$ (i.e., $\chi_{ij} \in \mathbb{Z}_2$ for each $1 \leq i \leq m$, $1 \leq j \leq n$).

$\mathfrak{B} = \{X_{b_i^{j_i}}\}$ is a finite generator of χ acting as a simplicial complex for image b_i. More generally, if k is a commutative ring with unit then, any $\chi \in M_{m \times n}(k)$ can be expressed as a sum of the form:

$$\chi = \sum_{j=1}^{k} \sum_{i=1}^{h} E_{ij} \otimes \chi_{ij} \tag{3.3}$$

where $E_{ij} \in M_{n \times m}(k)$ is an unitary matrix, and for each i, j, $\chi_{ij} \in M_{n_i \times m_j}(k)$ is associated to a model of an image repository $\mathcal{B}_{ij}$ with
$\sum\limits_{i=1}^{k} n_i = n$ and $\sum\limits_{j=1}^{h} m_j = m$.

This setting implies that each matrix χ associated to a model of an image repository is a mosaic.

Observe that $\mathfrak{B}$ is a finite generator of χ such that:

$$\left.\begin{array}{rll} X^r_{b_i} \cap X^s_{b_i} = & \varnothing \quad \text{if} \quad r \neq s \\ X^m_{b_i} \cap X^n_{b_j} = & \varnothing \quad \text{for all} \quad i,j,m,n, \\ b_i = & \sum\limits_{j_i=1}^{t_i} a_{j_i} X^{j_i}_{b_i} \quad \text{for any } i, \quad a_{j_i} \in \mathbb{Z}_2, \quad 1 \leq i \leq k. \end{array}\right\} \tag{3.4}$$

The identities (3.4) allow us to write χ in the form:

$$\chi = \sum_{h=1}^{k}\sum_{j_h=1}^{t_h} a_{j_h} X^{j_h}_{b_h} \qquad , a_{j_h} \in k, \tag{3.5}$$

and thus, $\mathfrak{B}$ is a *tessellation* of χ induced by $\mathcal{B}$.

Definition 3.1.1. If $b_i \preceq b_j$ then b_j can be written in the form

$$b_j = b_i + W_j,$$

where W_j is a complementary subspace of b_i in b_j. In such a case, a chain $C = b_i = b_{i_1} \prec b_{i_2} \prec \cdots \prec b_{i_{j-1}} \prec b_{i_j} = b_j$,

with $b_{i_j} = \sum\limits_{h=1}^{j-1} b_{i_h}$, defines b_j as an emerging image. Therefore, an emerging image can be defined by a k-linear representation of a finite chain.

Definition 3.1.2. If $\mathfrak{B}' = \{b'_1, b'_2, \ldots, b'_s\}$ is an ordered database of binary images, and for some h with $1 \leq h \leq s$, there exists $b_r \in \mathfrak{B}$ such that

$$b'_h = \sum_{j_r=1}^{t_r} a_{j_r} X^{j_r}_{b_r}, \tag{3.6}$$

then the system of binary images (b_r, b'_h) is a multistable image.

This definition can be generalized by considering a collection $\mathfrak{B} = \{\mathfrak{B}^h : 1 \leq h \leq p\}$ of binary images. Thus, if for each fixed $\mathfrak{B}^h$, it is defined a model $\nu_h : \mathfrak{B}^h \to M$ with M a fixed k-module.

Thus, a system of images (ν_h, ν_i) is a multistable image provided that there exists $v \in M$ such that $\nu_h(I) = \nu_j(I') = v$ for some images $I \in \mathfrak{B}^h$, $I' \in \mathfrak{B}^j$.

If there exists a polynomial-time algorithm (or function) $\mathcal{A}$ such that for some positive integer n, $\mathcal{A}(n) = \chi$, then we say that n *encodes* or *encrypt* the repository $\mathcal{D}$.

3.2 Algorithm ATMMEI

Algorithm to Model Multistable and Emerging Images ATMMEI allows to obtain models of type $\nu(I)$ from a binary images database. Actually, the procedure can be generalized to gray-level and RGB images.

Algorithm 3.2.1: Algorithm ATMMEI

1. Choose a binary images Repository $\mathcal{D}$.

2. Define $M = M_n(\mathbb{Z}_2)$ a $\mathbb{Z}_2$-vector space of matrices with binary entries and a suitable basis $\mathcal{B} = \{b_i \mid i \in S\}$, where S is a fixed finite set of indices. $b_i \cap b_j = \varnothing$ if $i \neq j$. That is, distinct elements in $\mathcal{B}$ have no common entries.

3. Define a matrix representation $\chi \in M$ de $\mathcal{P}$ as follows:

   (a) Choose a suitable large positive integer $n = t_1 \times t_2$, a corresponding $t_1 \times t_2$-rectangular array $\mathcal{A}$ for its digits and a security parameter t.

   (b) Choose a sequence $\mathcal{S}$ of consecutive pairs of blocks (B_i, B_{i+1}), $B_i = (n_{i_1}, n_{i_2}, \ldots, n_{il_i})$, $B_{i+1} = (n_{(i+1)(1)}, n_{(i+1)(2)}, \ldots, n_{(i+1)l_{(i+1)}})$, $n_j \in \mathcal{A}$, $i \leq j \leq l_{(i+1)}$, $n_{i1}, n_{(i+1)1} \neq 0$, $l_i + l_{(i+1)} = t$. t is the minimum positive integer such that:

   $$\sum_{h=1}^{l_i} n_{ih} = \sum_{m=1}^{l_{i+1}} n_{(i+1)m}.$$

   (c) Construct χ as a $t_1 \times t_2$-bitmap via the function $\gamma : \mathcal{S} \to \mathbb{Z}_2^t$ defined in such a way that $\gamma(B_i, B_{i+1}) = (c_i1, c_{i2}, \ldots, c_{il_i}, d_{(i+1)1}, d_{(i+1)2}, \ldots, d_{l_{(i+1)l_{(i+1)}}})$, where $c_{ij} = 1$ and $d_{(i+1)s} = 0$, $1 \leq j \leq l_i$, $1 \leq s \leq l_{(i+1)}$.

4. Write $\chi = \sum\limits_{i \in S} b_i$.

5. For each image $I \in \mathcal{D}$ write $\nu(I) = \sum\limits_{h \in T \subset S} b_h$,

   i.e., each image $\nu(I)$ is obtained from χ by applying on it admissible transformations.

The mathematical model permits to obtain the following important result.

Theorem 3.2.1. *There is an infinite number of emerging images induced by digits of positive integers.*

Proof. Let $\mathfrak{M} = (\mathcal{B}, \chi, \mathfrak{B}, \nu)$ be a fixed system of models arising from a database of binary images $\mathcal{B}$, where χ is a vector of the vector space $M_p(\mathbb{Z}_2)$ generated by Algorithm 3.2.1 with digits of the decimal expansion of an irrational number r, $p = t_1 \times t_2$, $\mathscr{B}$ is a generator of χ and ν is a model defined on $\mathcal{B}$.

Let $\{\mathfrak{M}_i \mid 1 \leq i \leq n^2\}$ be a sequence of models with $\mathfrak{M}_i = (\mathcal{B}_i, \chi_i, \mathfrak{B}_i, \nu_i)$, $\mathfrak{M}_i = \mathfrak{M}$, $\chi_i \in M_p(\mathbb{Z}_2)$, $\chi_i \neq \chi_j$ if $i \neq j$. Then for a database $\mathcal{D}$, $\mathcal{B} \subset \mathcal{D}$, we define a system $\mathfrak{N}$ of models defined by $\mathcal{D}$ in such a way that $\mathfrak{N} = (\mathcal{D}, \psi, \mathfrak{D}, \delta_1)$, where $\psi = \bigoplus_{i=1}^{n^2} \chi_i$ is a $n^2(t_1 \times t_2)$-bitmap generated by Algorithm ATMMEI 3.2.1 with $n^2(t_1 \times t_2)$ consecutive digits of r. In this case, $n\mathfrak{B} \subset \mathfrak{D}$, i.e., magnifications by a factor n of elements of $\mathfrak{B}$ are used to generate ψ and thus if $\nu(I)$ is a model of an image I then $\delta_1(\nu(I))$ is a model of the magnification nI of I. Since this process can be repeated infinitely many times with all images $I \in \mathcal{B}$ modeled by ν, that is, $\ldots \delta_k(\ldots(\delta_2(\delta_1(\nu(I))))\ldots)$ is a sequence of distinct emerging images generated by I, then there exists an infinite number of images of this type. This proves Theorem 3.2.1. $\square$

Remark 3.2.2. A model $\mathfrak{G}$ of a binary image repository $\mathscr{D}$ is said to be *generated* by a matrix representation M of a Poset $\mathcal{P}$, provided that $\chi \in \mathcal{O}(\mathcal{M})$.

Remark 3.2.3. Algorithm 3.2.1 also works when the sequencing process in steps (b) and (c) is modified by determining the different locations of the first digit n_1 in the large integer n and assigning a binary pixel value. In this thesis, we assign pixel value 0 to all digits $d \in n$ in an interval $F_1 \leq d < F_0$, where $F_i, i \in \{0, 1\}$ denotes consecutive occurrences of the first digit n_1 and F_0 is an odd location of this number.

3.3 Multistable and Emerging Images-based HIPs

The proposed model for creating emerging and multistable images and the automated solution of HIPs is described in Algorithm 3.2.1.

It allows us to construct artificial patterns with digits in the decimal expansion of some irrational numbers. In this work, we use questions in the HIPs posed by ARTiFACIAL, BONGO and reCAPTCHA with the following characteristics:

1. **ARTiFACIAL**: we ask to the user to interpret a given model in order to locate, eyes, nose, mouth, hands and feet by clicking on some points of the image.
2. **BONGO**: An user must to identify the difference between two blocks of emerging or multistable images.
3. **reCAPTCHA**: An emerging-text resistant to segmentation must be recognized by an user. The disadvantage observed in this case is that sometimes is very tricky to identify the proposed text.

In the experimental results we show the HIP used. Furthermore, processes and algorithms to extract images are presented in next chapters.

Chapter 4

Emerging Images Associated to Tiled Orders and Posets

In this chapter, we use 01-tiled orders or semimaximal rings to generate models of images associated to a repository $\mathcal{D}$. We also define emerging equipped posets, whose structure allow us to order images arising from different repositories and encoded by a Brauer configuration associated to large integer number.

This chapter also describes the system REIADT (Reconstruction of Emerging Images by Restricted Admissible Transformations) which is an algorithm to efficiently extract images from master shares.

Initially we analyzed the way that 01-tiled orders allow us to interpret images as posets. This result is very relevant and will be explained as follows.

Any 01- tiled order has an associated finite poset $\mathcal{P}$, such that if $\Lambda = (a_{i,j})$, then

$$a_{i,j} = \begin{cases} 0, & if \quad i \leq j, \\ 1, & otherwise. \end{cases}$$

Example 4.0.1. Let Λ be the 01-tiled order

$$\Lambda = \left(\begin{array}{ccccc} & a_1 & a_2 & a_3 & a_4 \\ a_1 & 0 & 1 & 0 & 0 \\ a_2 & 0 & 0 & 0 & 0 \\ a_3 & 1 & 1 & 0 & 0 \\ a_4 & 1 & 1 & 1 & 0 \end{array} \right),$$

where 1 indicates that there exists connection and 0 indicates otherwise, such that, the first are comparable points and last are incomparable points.

The Poset $\mathcal{P}_\Lambda$ associated to Λ is presented in Figure 4.1.

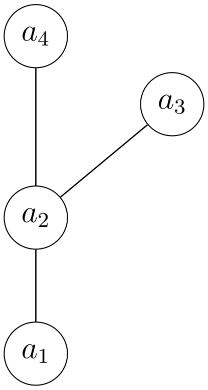


Figure 4.1: Poset $\mathcal{P}_\Lambda$ associated to the 01-tiled order Λ

Example 4.0.2.

From 01-tiled order Λ_1 can be constructed subposets, which is very important when we associate them with the images.

$$\Lambda_1 = \left(\begin{array}{ccccc} & a_1 & a_2 & a_3 & a_4 \\ a_1 & 0 & 0 & 0 & 0 \\ a_2 & 0 & 0 & 0 & 0 \\ a_3 & 0 & 1 & 0 & 0 \\ a_4 & 1 & 0 & 0 & 0 \end{array}\right)$$

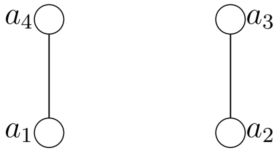


Figure 4.2: Poset $\mathcal{P}_{\Lambda_1}$ associated to 01-Tiled order Λ_1

Example 4.0.3.

In Figure 4.3, an example about different subposets from 01-tiled order Λ_1 is presented.

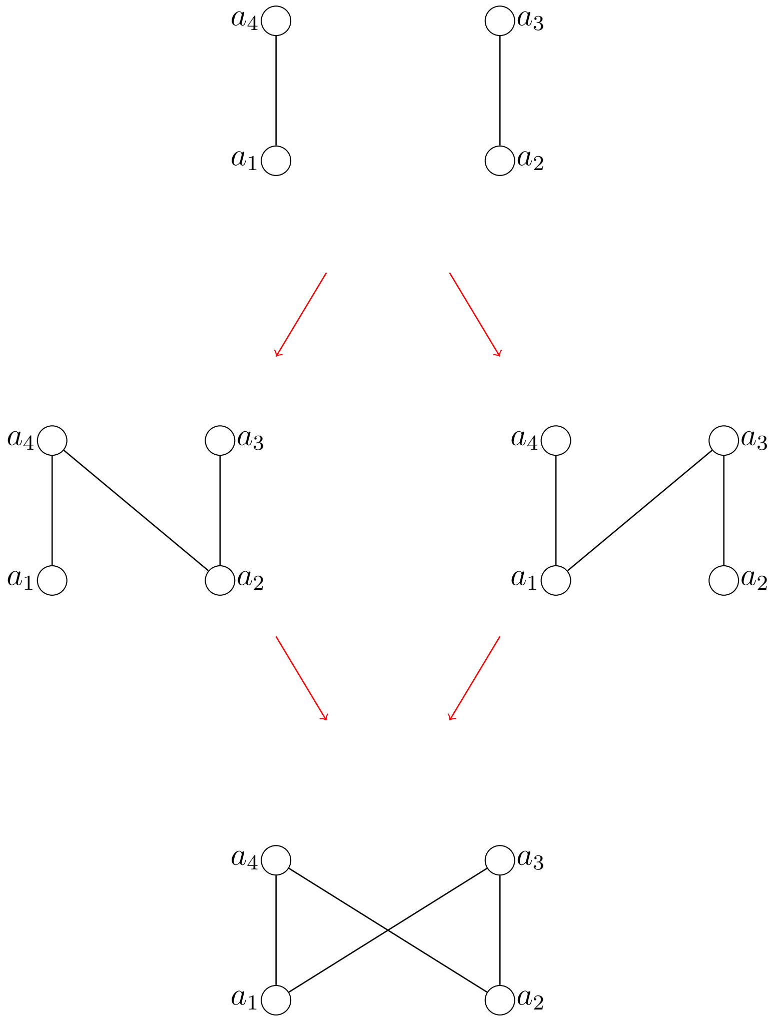


Figure 4.3: Example of subposets generation

The following results show the way of generating emerging images from a repository $\mathcal{D}$.

Theorem 4.0.4. *Let $\mathcal{D}$ be an image-repository and Λ be a* 01*-tiled order. Then Λ models $\mathcal{D}$ provided that any element $I \in \mathcal{D}$ can be represented as an element of Λ.*

Proof

Let $I \in \mathcal{D}$ be an image. The corresponding associated polygon $\mathcal{P}_I$ has the form

$$\mathcal{P}_I = \bigcup_{i=i_0}^{t} \bigcup_{h=0}^{k} [a_i, a_{j_h^i}],$$

where each point x is an interval $[a_i, a_{j_h^i}] \in \mathcal{P}_\Lambda$. In fact, the intervals allow us to establish where the portion of the restricted image to the ith row begins and ends.

The model $\nu(I)$ is obtained from $\mathcal{P}_I$ by applying suitable completions. Finally, I is calculated from $\nu(I)$ through a filtration process. This finishes the proof of Theorem 4.0.4 $\square$.

Remark 4.0.5. Any element of $\mathcal{D}$ is a linear combination of basic elements of a finitely generated Λ-module.

In case of a repository $\mathcal{R}$ is modeled in the sense of M.A.O. Angarita [26] by using a 01-tiled order, then each model $\nu(I)$ of an image can be viewed as a *subposet* of $\mathcal{P}_\Lambda$ associated to Λ.

We will say that the model $\nu(I)$ is a *polygon* associated to the image I. Therefore, each model $\nu(I)$ can be extracted from $\mathcal{P}_\Lambda$ by the following steps.

1. Polygon $\mathcal{P}(I)$ identification.

2. Noise elimination.

Polygon $\mathcal{P}(I)$ identification allows us to give information about the edge of $\nu(I)$, while the process of completion eliminates noise of salt and pepper type, which is included in $\nu(I)$. That is, by elimination of some relations in $\mathcal{P}(I)$.

Remark 4.0.6. The noise is constructed in the model to increment the capability of storing (so that, the number of images that can be modeled by Λ).

A polygon is defined by the following expression:

$$\mathcal{P}(I) = \bigcup_{i=i_0}^{t} \bigcup_{h=0}^{k} [a_i, a_{j_h^i}],$$

such that, all images can be seen in this way.

We have the union of intervals as follows.

$$\begin{aligned}
a_1 &= [a_1, a_{j_1^1}] \cup [a_1, a_{j_2^1}] \cup \cdots \cup [a_1, a_{j_k^1}],\\
a_2 &= [a_2, a_{j_1^2}] \cup [a_2, a_{j_2^2}] \cup \cdots \cup [a_2, a_{j_k^2}],\\
\vdots \quad & \quad \vdots \qquad\qquad \vdots \qquad\qquad\qquad \vdots\\
a_i &= [a_i, a_{j_1^i}] \cup [a_i, a_{j_2^i}] \cup \cdots \cup [a_i, a_{j_k^i}],
\end{aligned}$$

Let Λ be the 01- tiled order, which is composed by Λ_1 and Λ_2, that represent chains and they are defined as follows.

$$\Lambda_1 = \begin{pmatrix}
0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\
1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\
0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\
0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 \\
0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\
1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\
1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\
1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \\
0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\
0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\
1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\
1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\
1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \\
1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 \\
1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 \\
1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\
1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\
1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\
1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\
1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}$$

Figure 4.4: Tiled Order Λ_1 associated to scheme 1

$$\Lambda_2 = \begin{pmatrix}
0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 \\
1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\
1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \\
1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \\
1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\
1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\
0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 \\
0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\
1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\
1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 \\
0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \\
1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \\
0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\
0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\
0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\
1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\
1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\
0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\
1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\
0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\
0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\
1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\
1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\
1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\
1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\
0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\
1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0
\end{pmatrix}$$

Figure 4.5: Tiled Order Λ_2 associated to scheme 1

The chains Λ_1 and Λ_2, previously presented, are suitable transformed to obtain Λ_3 and Λ_4 to generate the corresponding bitmap.

So that, the generated bitmap or scheme 1 (Figure 4.6) will be used to obtain experimental results from the proposed algorithms.

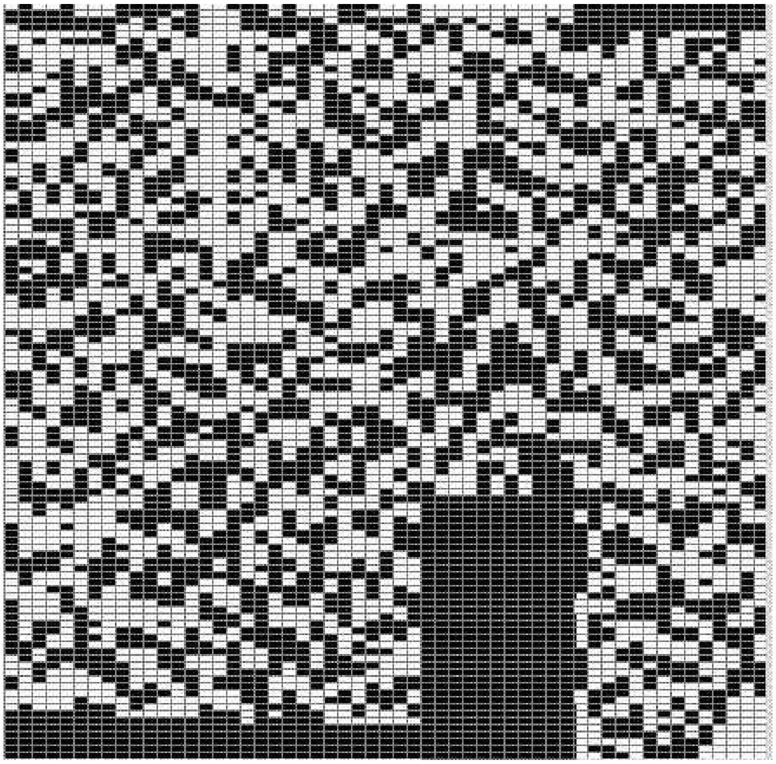


Figure 4.6: Bitmap associated to 01-tiled order.

Since the poset $\mathcal{P}_\Lambda$ has many features, it follows that its representation is not easy to obtain.

In the following section, the algorithm that allows us to extract the images from the bitmap is described in detail.

4.1 Generation of Emerging Images via Tiled Orders

From a 01-tiled already established the corresponding bitmap is then generated. For example, the presented scheme in Figure 4.6 is a matrix of 1's and 0's, so that, it is a subposet of 01-tiled order. Thus, by performing different stages described below, we will obtain images that have already been set by default in the repository $\mathcal{R}$.

Definition 4.1.1. An image repository $\mathcal{R}$ is said to be associated to a 01-tiled order if all of its images can be obtained after applying some restricted admissible transformations.

In the next section, we present an algorithm to establish the bitmap region and to extract the emerging images. This algorithm is contained in system REIADT.

4.2 Algorithm to generate Emerging Images ATGEI

The objective of this procedure is to obtain the border of each image that generates the bitmap of size $m \times n$.

Let X_k a binary image that has a set of ordered couples of pixels associated, which are not necessarily different.

The algorithm for generating emerging images has two main stages:

Algorithm 4.2.1: Algorithm ATGEI

```
1. Region recognition.

2. Image extraction.
```

4.2.1 Region recognition

Region recognition is the process for which a subset $\mathcal{I}$ of points is chosen from the poset $\mathcal{P}$ associated to the 01-tiled. $\mathcal{I}$ contains the pattern-image T which can be obtained after some restricted transformations. The points $m \in \mathcal{I}$ are called *markers*, which are denoted as $\vec{m}_{i,j}$, $\overset{\leftarrow}{m}_{i,j}$, $\overset{\uparrow}{m}_{i,j}$ or $\underset{\downarrow}{m_{i,j}}$. In this case, the arrows point out for which entries in columns or rows of the tiled order must be changed to extract the image.

These markers determine a sequence $M = \{m_{i_1,j_1}, m_{i_2,j_2}, \ldots, m_{i_k,j_k}\}$, which detects the pattern-image boundary. To do the pattern-image recognition, the procedure that is described in Algorithm 4.2.2 below is carried out.

Algorithm 4.2.2: Algorithm

1. If $m_s = m_{i_s,j_s}$ and $m_t = m_{i_t,j_t}$ are consecutive markers with $i_s < i_t$, then both m_s and m_t belong to the boundary of T and have pixel value $p(m) = 1$.

2. Compute the l_1-distance $d = d(m_s, R^w_{i_s+1})$ and let m_{i_s+1,j_k} the pixel or block of pixels such that $d = d(m_s, R^w_{i_s+1})$ and m_{i_s+1,j_k} belongs to the boundary of T. This process continues until the marker m_{i_t,j_t} is reached.

3. If there are two pixels or block of pixels m, m' such that $d = d(m_s, m) = d(m_s, m')$, then the vertex in the direction determined by the marker must be chosen.

Remark 4.2.1. In the boundary detection process there are some pixels or block of pixels belonging to the boundary of T with pixel value $p(m) = 0$. Such kind of pixels are said to be *bridges* with the main role of preserving the visual connectedness of the boundary. Formally, the boundary of T is an union of components determined by the markers.

4.2.2 Image extraction

Image extraction is the process that allows to construct in detail the emerging image. In the image extraction stage the pattern-image is obtained after application of the following *restricted admissible transformations*:

1. row and column permutations;
2. additions of entries in columns at the left (right) of $\overleftarrow{x_i}$ ($\overrightarrow{x_i}$);
3. additions of entries in rows above (below) of $\overset{\uparrow}{x_i}$, $(\underset{\downarrow}{x_i})$;
4. completion (changing the value of an entry);
5. image rotation.

The Completion step is a way of reducing the width of the poset associated to 01- tiled order. It is applied to the pixels within the region determined by the markers. Via rotation, the detection boundary algorithm is defined similarly for any class of markers.

As an example, in the Figure 4.7, we show the images associated to an array induced by a 01 tiled order.

A. Zavadskij [99] defined the completion procedure, (i.e. by adding a suitable relation between two special points a and b) as follows.

The completion or (a, b)-completion of the set Q with respect to the special pair (a, b) consists in joining to Q the relation $a < b$. The obtained completed poset is denoted by $\overline{Q}_{(a,b)}$.

In this research we present the completion to any point in the poset, which differs from that due to Zavadskij who defined the completion as a process of adding relations between special points.

Figure 4.8 shows the filtered image, which can be used in HIPs.

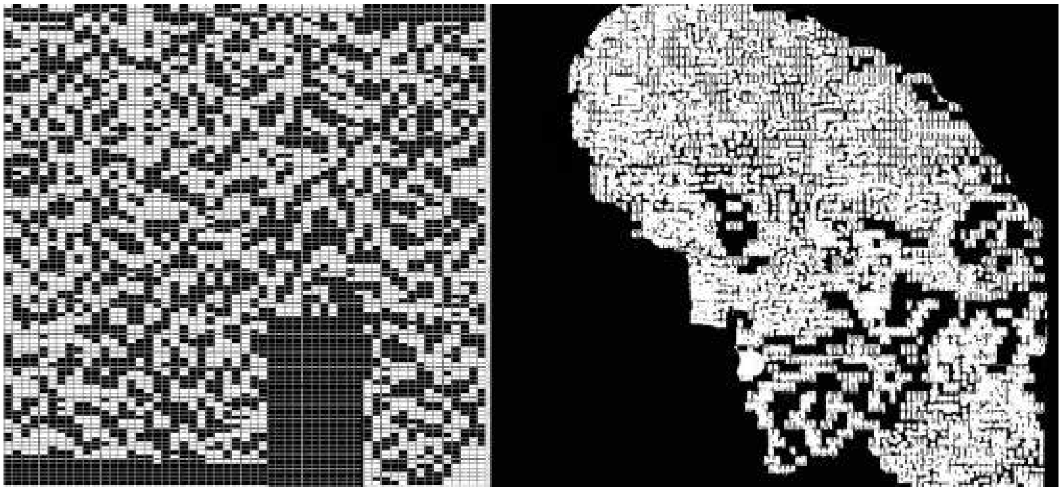


Figure 4.7: Subset of points induced by a 01−tiled order and image obtained after an extraction process.

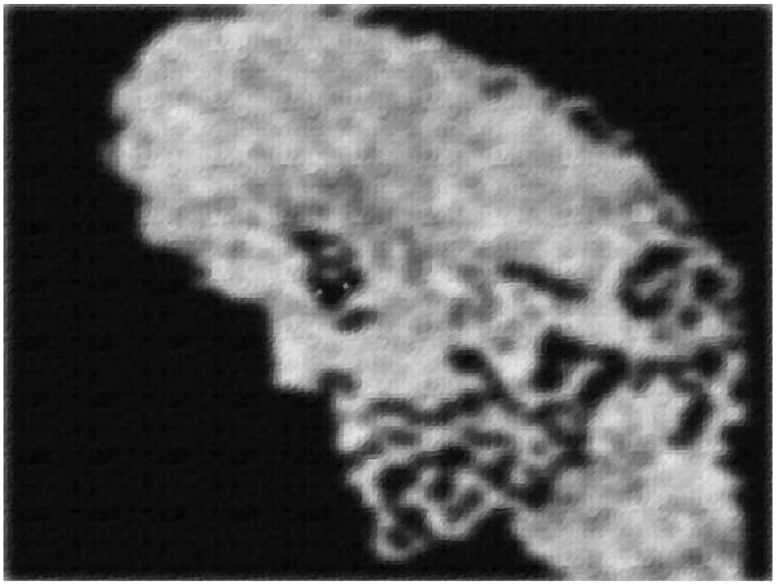


Figure 4.8: Bitmap associated to 01-tiled order.

For the development of the Algorithm 4.2.2, three processes are required:

1. Movement.

2. Appendices elimination.

3. Pruding.

Movement

In this stage the Algorithm 4.2.2 goes through the array from the markers and it selects a temporal border for the image X_k. We denote it by X'. In this case, is enough to describe the behavior of the algorithm in vertical direction. We assume the bitmap $a(X)$ is divided in the blocks B_i, as mentioned below.

These blocks are constituted by m_i arrows, listed below upwards and n_i columns listed from left to right. By the features of this algorithm every stage will be realized block by block from the corresponding marker.

Definition 4.2.2. $X \subseteq a(X)$ is arc-connected if every pair of points $a_i, a_j \in X$, with pixel values $p(a_i) = p(a_j) = 0$ can be connected by means of a permissible path σ with the property that if $a_t \in \sigma$, then $p(a_t) = 0$.

Remark 4.2.3. The path σ, with initial point a_i and final point a_j, can be written as $\sigma = \overline{a_i a_j}$; the set of admissible paths that connect a_i with a_j is denoted S.

Definition 4.2.4. If d is the distance of the taxicab and $\sigma \in S$, then the length of σ, is denoted by $\|\sigma\| = d(a_i, a_j)$.

Remark 4.2.5. Let $a(X)$ a rectangular bitmap, $A \subseteq a(X)$ and $a_{ij} \notin A$ with pixel value $p(a_{ij}) = 0$. Then h_i is the element of A in the i-th arrow with pixel value $p(h_i) = 0$ such that $d(a_{ij}, A) = \|\sigma\|$ for some admissible path $\sigma, a_{ij}h_i$. If such σ does not exist, then $h_i \in A$ satisfies that $d(h_i, a_{ij}) = d(A, a_{ij})$.

The definitions and remarks above allow us to describe a movement in the i-th block in the following way.

Let m be a marker of X' in the position ij and R_k the k-th arrow of the k-th block. Then the generated set H on which the algorithm acts is

$$H = \{h_t : \quad h_t \in R_t, \quad i+1 \leq t \leq n_i\}.$$

If we denote $A(a_{ij}) = p(a_{ij})$ the output of algorithm in any of its stages, then during a movement if σ is $a_{ij}h_{i+1}$-admissible of minimum length, then for each $a_{ik} \in \sigma$, whenever pixel value $p(a_{i+1k}) = 0$ and the curve $A(\sigma)$ is admissible, then:

$$A(a_{ik}) = \begin{cases} p(a_{ik}), & j \leq k \qquad or \quad p(a_{ik}) = 1, \\ p(a_{ik}) + 1, & p(a_{ik}) = 0 \quad and \quad k \lneq j. \end{cases}$$

Afterwards, we do the change $h_{i+1} \longleftrightarrow a_{ij}$ and the algorithm is applied to R_{i+1}.

In general, this condition has to be for $h_t, \quad i+1 \leq t \leq n_i, \quad h_t \longleftrightarrow a_{ij}$. The algorithm must be applied to each row of the i-th block .

Remark 4.2.6. *$H \subseteq X'$, for each $h_t \in H$ is associated to a curve $\sigma_t, a_{ij}h_t$-admissible, this curve is of lesser length such that $h_s \in \sigma_t, \quad i \leq s \leq t$. Admissibility also implies that the stretch of σ_t, which connects the points, h_{t-1} with h_t, corresponds to the form a_{t-1j} or a_{tj}, this implies that elements h_t are all contained in the same arc-connected component of X'.*

The elements $a_{ij} \in X'$ which are generated by a movement, are classified into three categories: bridges, appendices and growths. These categories are defined as follows.

Definition 4.2.7. A **bridge** is the element $a_{tj} \in b(X)$ where $h_t \in H$, $\delta =$ Min$\{d(h_t, h_{t-1}, d(h_t, h_{t+1})\} \gneq 2$ and $j =$Max$\{j_1, j_2\}$ if $j_i, i = 1, 2$ is the column that corresponds to element $h_{t\pm1}$ in the array.

A bridge changes temporally its pixel value to extend the arc-connectivity of the component that belongs to h_{t-1}.

Definition 4.2.8. An element $a_{ij} \in X'$ with pixel value $p(a_{ij}) = 0$ is an **appendix** if $p(a_{i+1j}) = p(a_{i-1j}) = 1$ and for a_{kj} the first in the jth column with pixel value $p(a_{kj}) \neq 1$, $Supp\{a_{lj} : i+1 \leq l \leq k\} \geq 2$ or a_{ij}, with pixel value $p(a_{ij}) = 0$ satisfies

$$a_{ij} \in \{a_{mj} \in b(X) : d(a_{mj}, X') \gneq 2\}.$$

Definition 4.2.9. An element $a_{ij} \in X'$ with pixel value $p(a_{ij}) = 0$ is a **growth** if $p(a_{ij+1}) = 0$.

Remark 4.2.10. The characterization given in the definition 4.2.7 is adjusted to the movement and the marker that are being taken from the beginning. Any another growth obtained from a different type of marker, is analogously defined.

Appendices and growths elimination

Eliminating appendices and growths is the appropriate way to obtain the definitive boundary X^n of a binary image generated by the system REIADT.

Appendices elimination

If n_i is the number of arrows in the i-th block and L y C are the set of appendices and growths respectively, which are obtained by one movement, then for each $a_{ik} \in X'$,

$$A(h_t) = \begin{cases} p(a_{ik}), & if \quad a_{ik} \notin L \cup C, \\ p(a_{ik}+1), & if \quad a_{ik} \in L. \end{cases} \tag{4.1}$$

Remark 4.2.11. If the position of $h_t \in H$ generates a bridge in the places: a_{tj_0}, h_{t-1} , a_{t-1,j_1} and h_{t+1} in the place a_{t+1j_2}, $j_o = Max\{j_1, j_2\}$, then $\{a_{t\pm 1j} : Min\{j_1, j_2\} \leq j \leq j_0\}$, are appendices.

Pruning (Growths elimination)

If $h_t \in H$ is a growth, then $A(h_t) = p(h_t) + 1$, whenever $p(h_t) + 1$ does not generate an appendix. Furthermore, if X'' is the obtained boundary from X' by the growth elimination, then $Supp(X'') <= Supp(X')$.

Once the algorithm is applied, it is possible to recover some growths that would allow a greater association of the obtained image by the algorithm from the original image.

The extracted image, presented in Figure 4.9, is an interpretation of the face of Jesus.

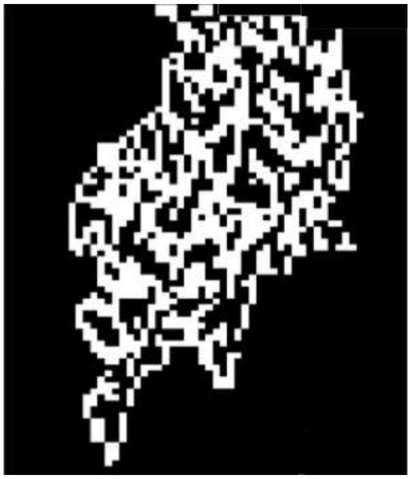


Figure 4.9: Extracted Image.

The markers and bridges used to obtain Figure 4.9 are:

${}_{\leftarrow}(21,0)$, ${}_{\leftarrow}(13,23)$, ${}^{\leftarrow}(13,47)$, ${}_{\leftarrow}(13,47)$,${}^{\leftarrow}(14,60)$, ${}_{\leftarrow}(14,60)$,${}^{\leftarrow}(14,76)$, ${}_{\leftarrow}(14,76)$, ${}_{\downarrow}(28,60)$, $(29,63)_{\rightarrow}$, ${}_{\downarrow}(28,60)$, $(31,63)^{\leftarrow}$, $(30,63)_{\leftarrow}$, $(46,24)^{\rightarrow}$, $(46,24)_{\rightarrow}$,$(30,64)_{\rightarrow}$, $(13,47)_b$, $(8,29)_b$, $(14,60)_b$, $(31,57)_b$, $(31,55)_b$, $(38,44)_b$, $(45,31)_b$, $(46,36)_b$, $(16,67)_b$, $(44,31)_b$.

Each arrow indicates how movement is applied. For example, ${}^{\leftarrow}$ indicates that pixels at the left of the markers will be eliminated, whenever b in subindex indicates there is a bridge in that position.

Figure 4.10 shows an example of a sequence of images obtained after applying Algorithm 4.2.2 from scheme 1.

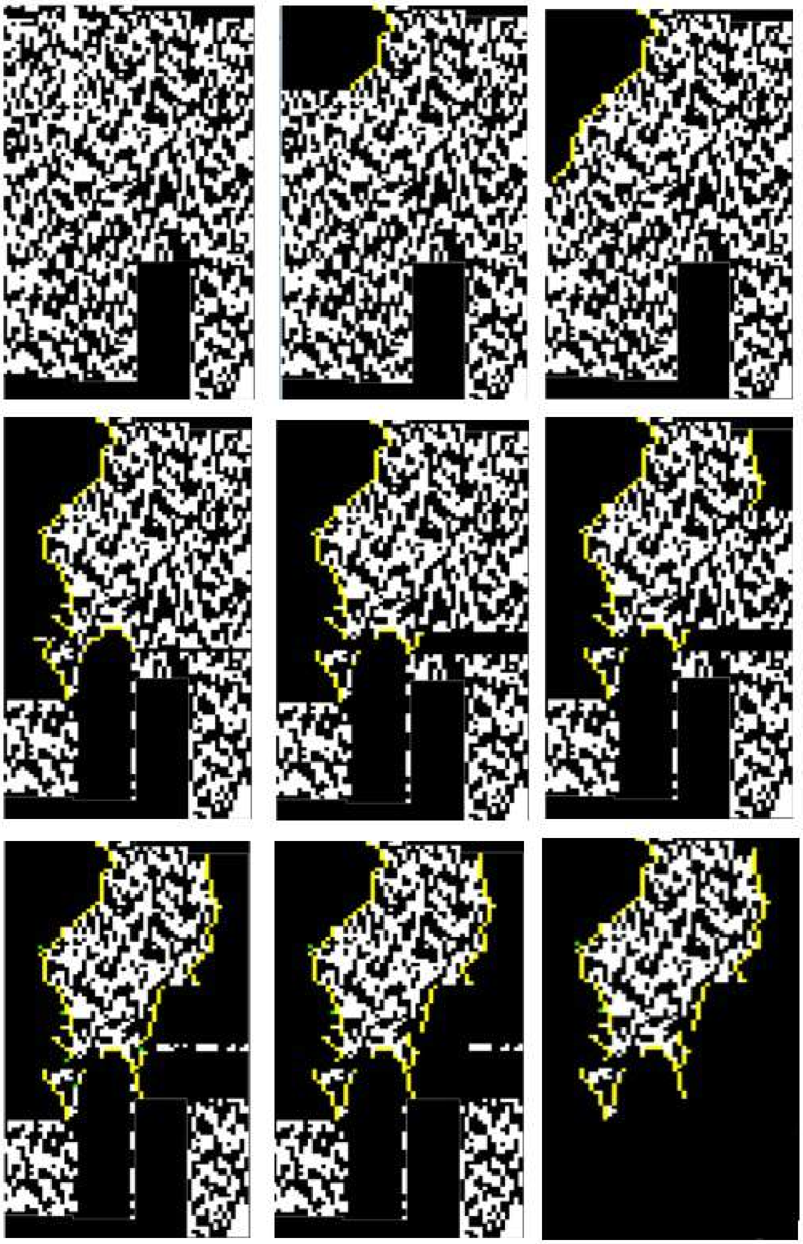


Figure 4.10: Sequence: Extraction of Images.

Theorem 4.2.12. *Poset $\mathcal{P}$ generates a repository $\mathcal{D}$, which means that all images in $\mathcal{R}$ can be described as a linear combination of subposets of $\mathcal{P}$.*

Proof

It is enough to use Algorithm 4.2.2 □.

Remark 4.2.13. A poset $\mathcal{P}$ is a sum of chains or subposets, which permits to have an interpretation of each image.

$$\mathcal{P} = a^{\triangledown} + b_{\vartriangle} + c : \mathcal{P} = \mathcal{P}_1 + \mathcal{P}_2 + \cdots + \mathcal{P}_t$$

In Figure 4.11 an example of subposet is presented.

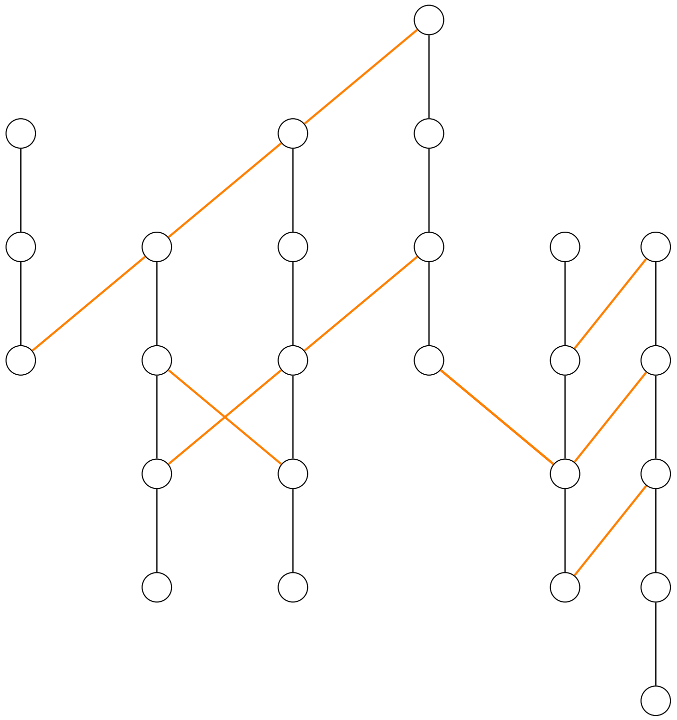


Figure 4.11: Example of subposet $\mathcal{P}_t$ associated to scheme showed in Figure 4.6

Remark 4.2.14. Some images of repository $\mathcal{R}$ correspond to the Davinci's drawings (see Figure 4.12), the images that will be extracted corresponds to interpretation of such drawings.

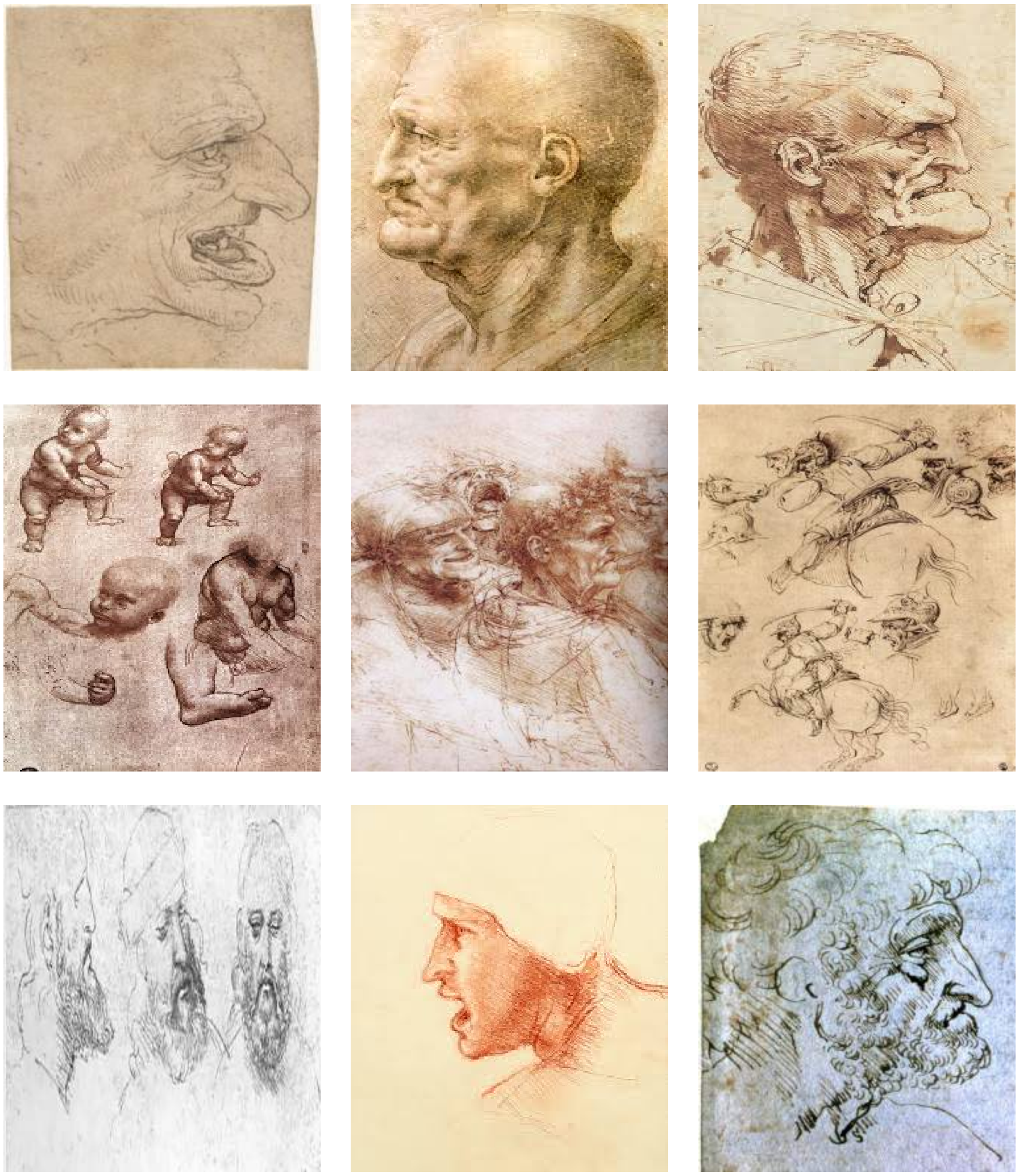


Figure 4.12: Davinci's Drawings.

All images contain ***noise***, which is understood as incomparable points in the poset. A *filter* to eliminate noise is interpreted as reduction of antichains in the associated poset.

Other images of repository $\mathcal{R}$ correspond to the Rembrandt's drawings (see Figure 4.13).

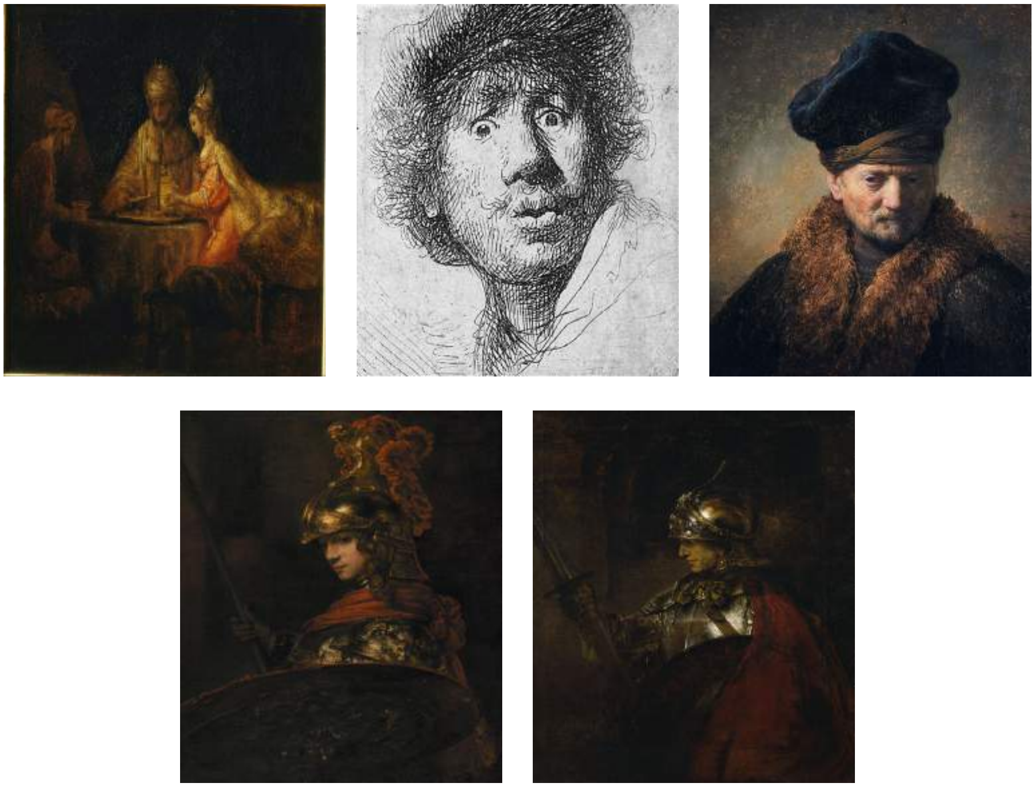


Figure 4.13: Rembrandt's drawings.

For instance, in Figure 4.14 the extracted images by using Algorithm 4.2.2 are presented. These images are extracted from others schemes, where the process of noise elimination has not been realized.

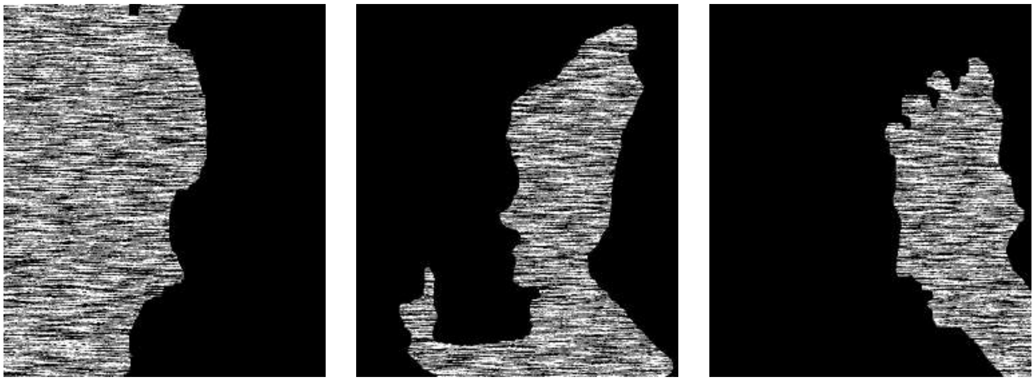


Figure 4.14: Extracted Images without noise elimination

In Figure 4.15, the extracted images by using Algorithm 4.2.2 are presented, although they were processed by reduction of antichains of poset, which means, the noise has been eliminated.

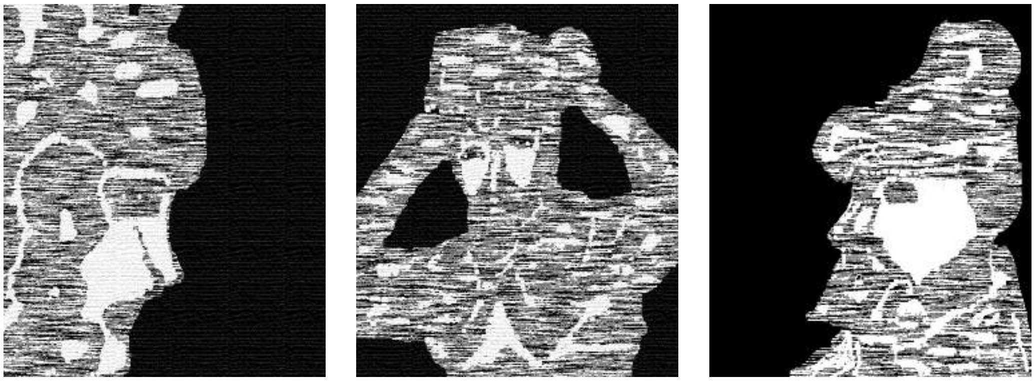


Figure 4.15: Extracted Images with noise elimination

The images of Figure 4.16 have been extracted by using Algorithm 4.2.2 and after a filter has been applied. In the Appendix, we show more examples of extracted images.

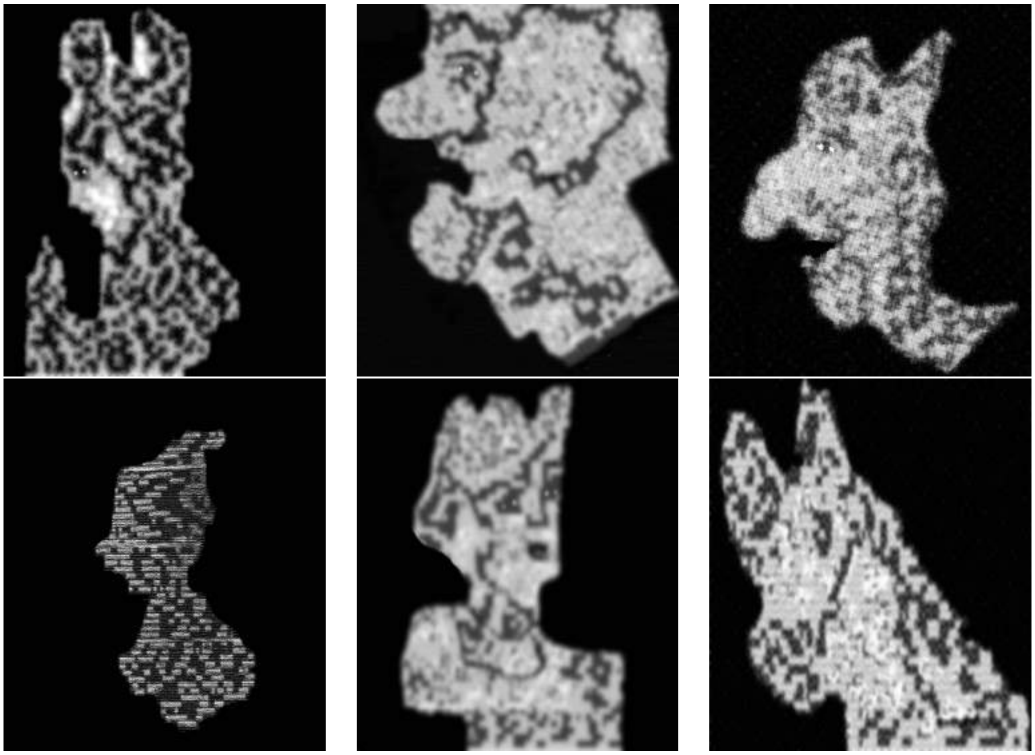


Figure 4.16: Examples of extracted images using algorithm 4.2.2.

Theorem 4.2.15. *The series of images type* 32 *(see Figure 4.17) are coded with the perfect number* $2^{756839-1}(2^{756839}-1)$.

Proof

From 01-tiled order is applied the algorithm 4.2.3 □.

Algorithm 4.2.3: Algorithm to generate images

1. Number sequencing.
2. Polygon-poset identification.
3. Noise filtering.

In the Figure 4.17, we present some images obtained by means of the Algorithm 4.2.3.

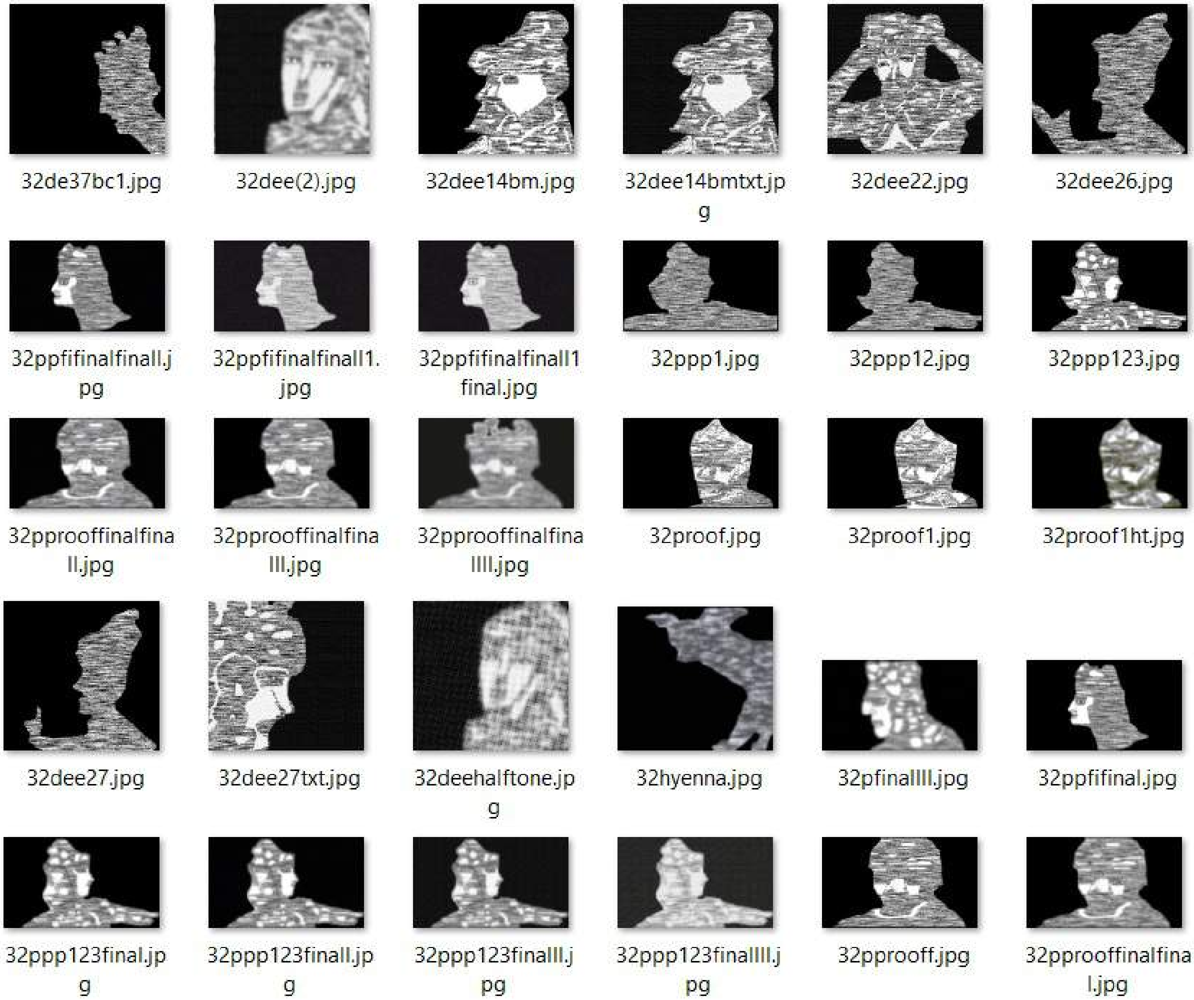


Figure 4.17: Series of images type 32.

4.3 Emerging Equipped Posets

Emerging Equipped Posets allows us to order images arising from different repositories encoded by a Brauer configuration associated to a large positive integer.

Definition 4.3.1. A common subject associated to a collection of image-repositories is said to be a *source*. Images given the same information in different repositories are said to be *profiles*.

For instance, a classical book or fairy tale can be considered a source whereas the different interpretations of a given personification can be considered profiles.

Definition 4.3.2. An Emerging Equipped Poset (EEP) denoted by $\mathcal{P}$ is a partially ordered set induced by sources and profiles as follows:

1. Points $x \in \mathcal{P}$ and relations are either weak or strong. In this case

   $\mathcal{P} = \mathcal{P}^{\circ} + \mathcal{P}^{\otimes}$.

2. Each weak point $x \in \mathcal{P}^{\otimes}$ has only one profile associated $\mathcal{D}_x$ and only one source associated.

3. Each strong point $x \in \mathcal{P}^{\circ}$ has several sources and several profiles associated.

4. If $x, y \in \mathcal{P}^{\otimes}$ then the relation between x and y is weak (denoted $x \preceq y$ provided that $\mathcal{D}_x \subseteq \mathcal{D}_y$.

5. If $x \in \mathcal{P}^{\otimes}$ and $y \in \mathcal{P}^{\circ}$ then the relation between x and y is strong (denoted $x \trianglelefteq y$). If the profile $\mathcal{D}_x$ is contained in the set of sources S_y associated to y. If $x, y \in \mathcal{P}^{\circ}$ then the relation between x and y is also strong with $S_x \subseteq S_y$.

Theorem 4.3.3. *1. For each point $x \in \mathcal{P}^{\otimes}$, $x \preceq x$ holds. In other words, each weak point has a weak relation with itself.*

2. Each strong point in an EEP is maximal.

Proof

1. By item 4 in definition 4.3.2, if $x \in \mathcal{P}^{\otimes}$, the relation between x and x is weak ($x \preceq x$) from $\mathcal{D}_x \subseteq \mathcal{D}_x$.

2. Let z be a strong point, so that, $z \in \mathcal{P}^{\circ}$. Assume that z is a strong point in an EEP, which is not maximal. Then there exists $m \in \mathcal{P}$ such that $m \trianglelefteq z$. There exists a weak point x such that y would have some strong points and some profiles associated. This is not possible, and therefore it is a strong point.□

4.4 The associated problem

In the construction of emerging images, we use information of digits of irrational numbers to generate schemes (see Figure 4.18). These schemes contain great quantity of valuable information.

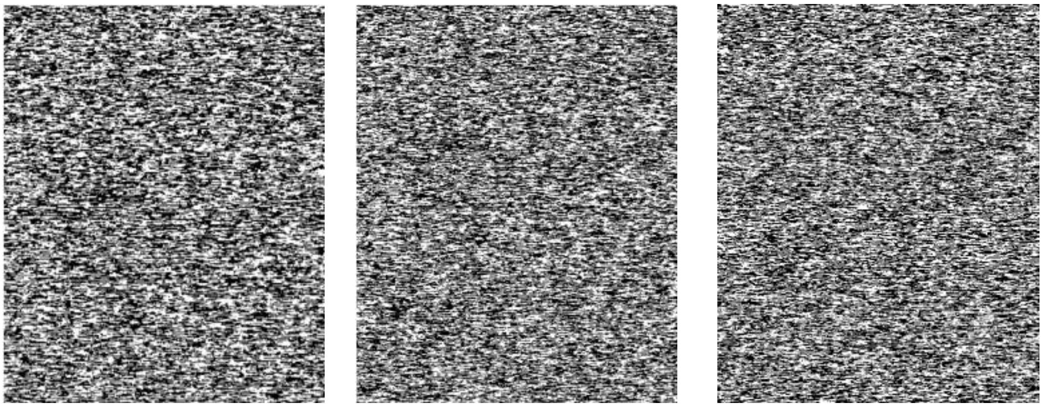


Figure 4.18: Examples of vectors $\chi \in M_{600}\ (Z_2)$.

According to Borwein et al. [21], within Number Theory, many problems offer large amount of data that the human mind has difficulty assimilating directly. These include distributions of digits in expansions. For instance, in the 17th Century, Gottfried Wilhelm Leibniz asked in a letter to one of the Bernoulli brothers if there might be a pattern in the binary expansion of π.

Three hundred years his question remains unanswered. A far the numbers in the expansion appear to be completely random. In fact the Leibniz's question can be generalized to all irrational numbers.

Research regarding digits of irrational numbers is related to that of *normal numbers*, where an irrational number x is said to be normal in base b if

$$\operatorname*{Lim}_{n\to\infty}\tfrac{B(n,j)}{n} = \tfrac{1}{b}$$

for each of the b possible values, $j = 0, 1, 2 \ldots b-1$, where $B(n, j)$ is the number of occurrences of j in the first n places of the b-ary expansion of x [16]. For instance, the Copeland-Erdös constant $C_{CE} = \sum\limits_{n=1}^{\infty} \frac{p_n}{10^{\sum_{k=1}^{n}\lfloor log_{10} p_k \rfloor + n}} = 0.23571113171923\ldots$ is normal for $b = 10$. So far, there is no proof of the normality of constants as π, e, $Ln2$, $\sqrt{2}$, or any irrational square root. In fact, these constants appear to be without pattern in the digits, and statistical tests done to date are consistent with the hypothesis that they are normal. Furthermore, according to Beyer et al. [16], no number has been proven to be absolutely normal, that is, normal in every base. Every normality proof so far is only valid in one based and depends on more or less artificial construction.

In order to give some advances to the Leibniz's question, we use digits of irrational square numbers, perfect numbers and Mersenne primes (i.e., primes of the form, $2^n - 1$) to construct emerging and multistable images. Such images allow us to define different kind of image based-HIP's in such a way that automatic solutions of these tests allow us to give a positive answer to the Leibniz's question.

Chapter 5

Multistable Images Associated to Brauer Configuration Algebras

In this chapter, we present a Brauer configuration associated to multistable images, from this mathematical structure allows a good representation of the information contained in these images. We explain how is the relationship between the Brauer configuration algebras and the process of extraction of these images.

5.1 Brauer Configuration Associated to Multistable Images

From a tessellation $\mathfrak{B}$ (see Figure 5.1) obtained by Cañadas et al. [26], we define its associated Brauer configuration $\Gamma_{\mathfrak{B}} = (\Gamma_{\mathfrak{B}_0}, \Gamma_{\mathfrak{B}_1}, \mu, \mathfrak{o})$, where:

- The vertices $v_x \in \Gamma_{\mathfrak{B}_0}$ are clusters of basic elements. In fact,

$$v_x = \sum_{J} cl_J(x).$$

- The polygons $V_x \in \Gamma_{\mathfrak{B}_1}$ consists of adjunct vertices.

If an image is in the repository $\mathcal{R}$ ($I \in \mathcal{R}$), then

$$V_I^1 < V_I^2 < V_I^3 < V_I^4 < \cdots < V_I^t = I, \text{ where } \quad V_I^r \in \mathcal{R}.$$

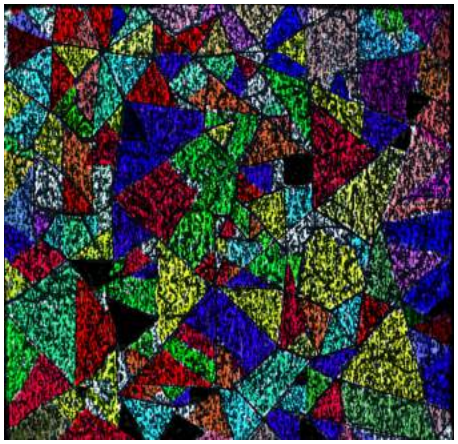


Figure 5.1: Tessellation [26]

Definition 5.1.1. An Image-Polygon is a polygon that has an interpretation in a repository $\mathcal{R}$ (see Figure 5.2).

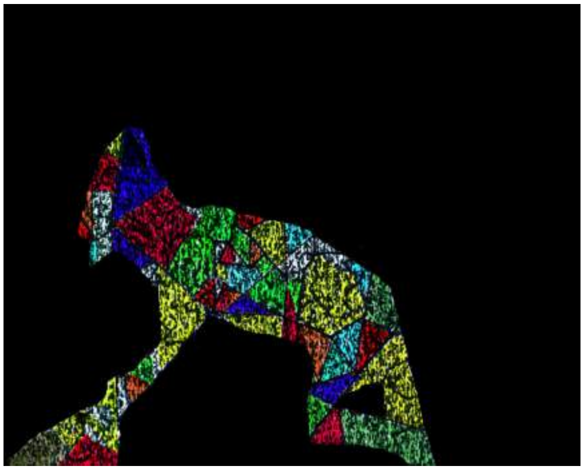


Figure 5.2: Example of Image-Polygon.

Remark 5.1.2. All polygons $V_x^I \in \mathcal{R}$ do not correspond to some image.

Remark 5.1.3. x is a sequence of polygons where V_x^i corresponds to **the parts** and V_I^r corresponds to **the whole** in the context of Emergence.

$$x :\Rightarrow V_I^{r_1} < V_I^{r_2} < \cdots < V_I^{r_s} = V_I^r.$$

Let χ be a representation of $\mathfrak{B}$, as we defined in Chapter 4. Then the following result is obtained.

Theorem 5.1.4. *The images obtained from χ by using Algorithm 3.2.1 are realizations of polygons defined by the following Brauer configuration:*

$$\begin{aligned}
\alpha^1_1 :&\quad x_{1,1} < x_{1,2} < x_{1,3} = x_{1,4},\\
\alpha^1_2 :&\quad x_{1,1} < x_{2,2} < x_{2,3} = x_{2,4},\\
\alpha^1_3 :&\quad x_{1,1} < x_{2,2} < x_{3,3} = x_{3,4},\\
\beta^2_1 :&\quad x_{2,1} < x_{1,2} < x_{1,3} = x_{1,4},\\
\beta^2_2 :&\quad x_{2,1} < x_{2,2} < x_{2,3} = x_{2,4},\\
\beta^2_3 :&\quad x_{2,1} < x_{3,2} < x_{3,3} = x_{3,4},\\
\gamma^3_1 :&\quad x_{3,1} < x_{2,2} < x_{2,3} = x_{2,4},\\
\gamma^3_2 :&\quad x_{3,1} < x_{2,2} < x_{1,3} = x_{1,4},\\
\gamma^3_3 :&\quad x_{3,1} < x_{3,2} < x_{3,3} = x_{3,4}.
\end{aligned}\tag{5.1}$$

Proof

Let the polygons be defined by

$x_{1,1} = \{\alpha^1_1, \alpha^1_2, \alpha^1_3\}$,
$x_{1,2} = \{\alpha^1_1, \beta^2_1\}$,
$x_{1,3} = \{\alpha^1_1, \beta^2_1, \gamma^3_2\}$,
$x_{1,4} = \{\alpha^1_1, \beta^2_1, \gamma^3_2\}$,
$x_{2,1} = \{\beta^2_1, \beta^2_2, \beta^2_3\}$,
$x_{2,2} = \{\alpha^1_2, \alpha^1_3, \beta^2_2, \gamma^3_1, \gamma^3_2\}$,
$x_{2,3} = \{\alpha^1_2, \beta^2_2, \gamma^3_1\}$,
$x_{2,4} = \{\alpha^1_2, \beta^2_2, \gamma^3_1\}$,
$x_{3,1} = \{\gamma^3_1, \gamma^3_2, \gamma^3_3, \}$,
$x_{3,2} = \{\beta^2_3, \gamma^3_3\}$,
$x_{3,3} = \{\alpha^1_3, \beta^2_3, \gamma^3_3\}$,
$x_{3,4} = \{\alpha^1_3, \beta^2_3\gamma^3_3\}$,

where $x_{i,j}$ correspond to the polygons and $\alpha^l_k, \beta^l_k, \gamma^l_k$ correspond to the vertices.

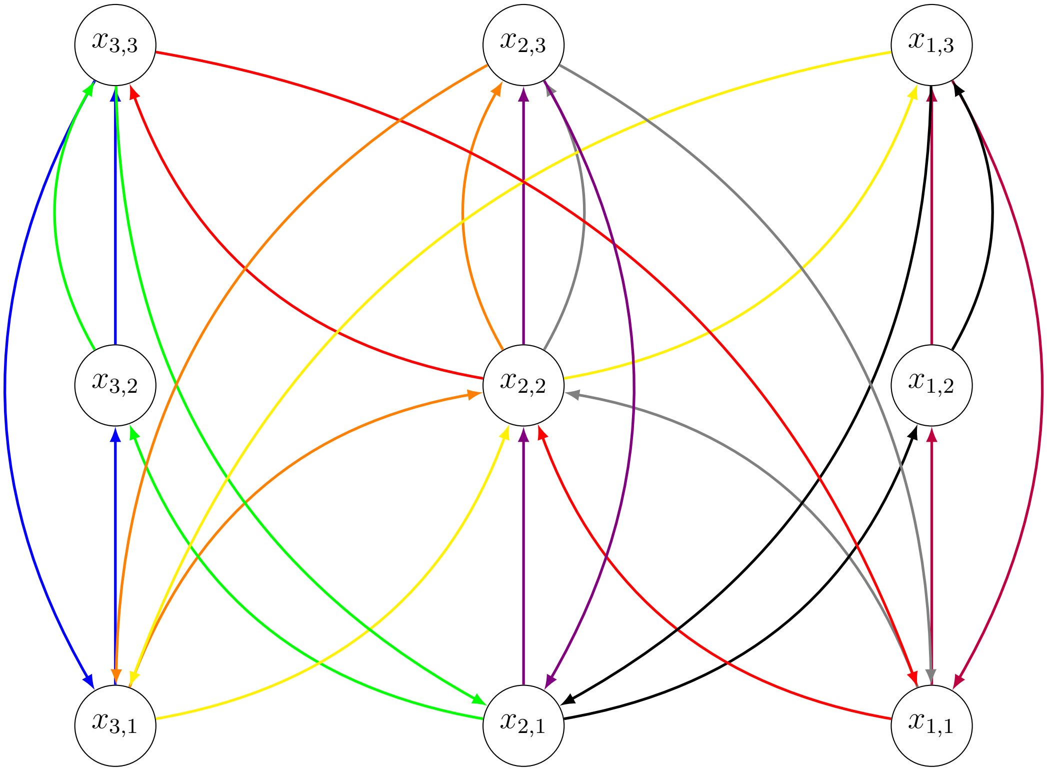


Figure 5.3: The quiver $Q_{\Gamma_{\mathfrak{B}}}$ associated to the Brauer configuration $\Gamma_{\mathfrak{B}}$.

The quiver $Q_{\Gamma_{\mathfrak{B}}}$ associated to the Brauer configuration $\Gamma_{\mathfrak{B}}$ is presented in Figure 5.3.

For the sake of clarity, we explicitly show the polygons $x_{1,4}, x_{2,4}, x_{3,4}$, which are maximal and they correspond to the Image-Polygons $\mathbf{I}_1, \mathbf{I}_2, \mathbf{I}_3$ (see Figure 5.4).

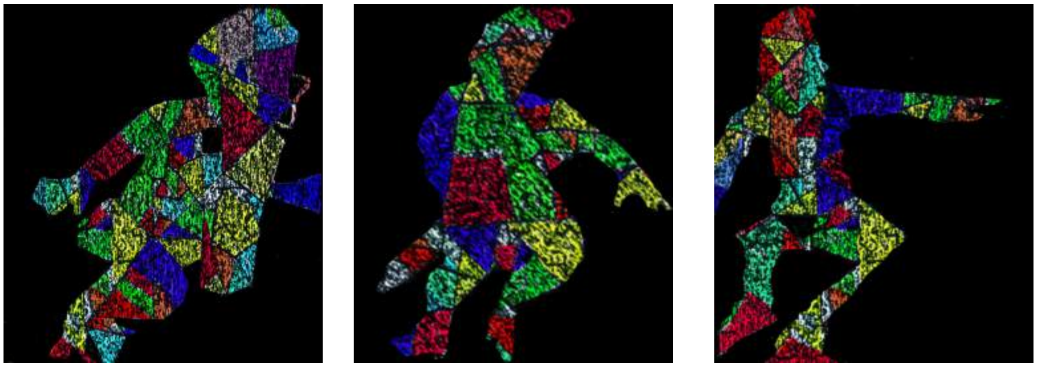


Figure 5.4: Example of maximal Image-Polygons

For example, $\alpha_1^1 = x_{1,1} < x_{1,2} < x_{1,3} < x_{1,4}$ and $\beta_2^2 = x_{2,1} < x_{2,2} < x_{2,3} < x_{2,4}$ can be described by the sequences of the Figures 5.5, 5.6, 5.7.

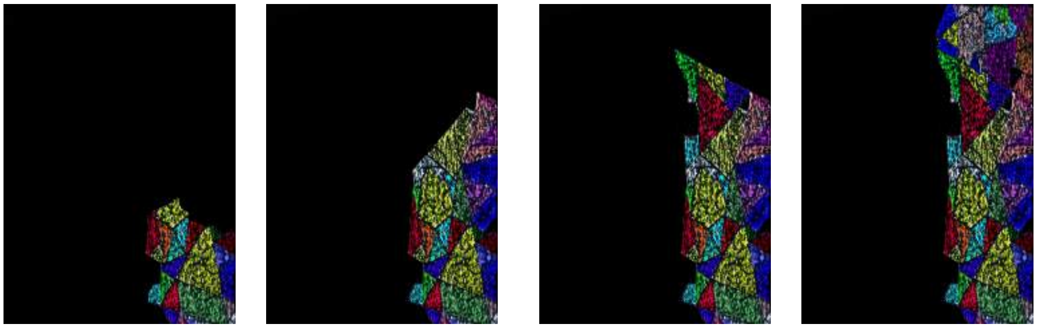


Figure 5.5: Example: Realizations of polygons α_1^1

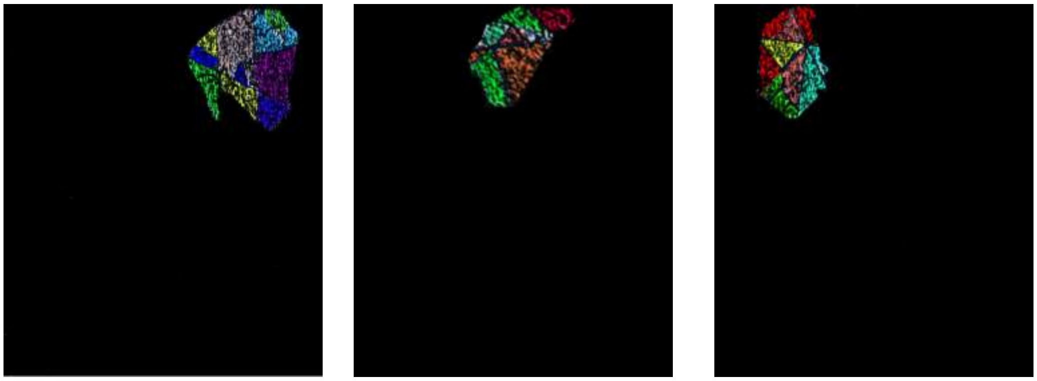


Figure 5.6: Example: Realizations of polygons γ_1^1

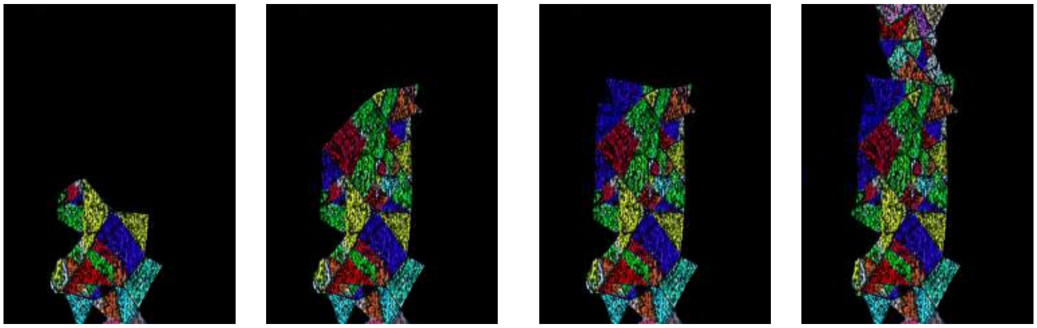


Figure 5.7: Example: Realizations of polygons β_2^2

The associated poset $\mathcal{P}_\Gamma$, which was used to define the Brauer configuration 5.1, is presented in Figure 5.8 $\quad\square$.

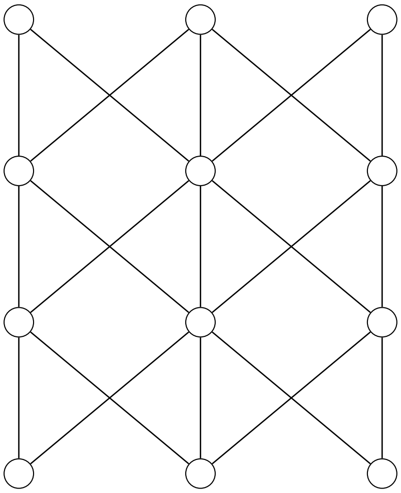


Figure 5.8: Poset $\mathcal{P}_{\Gamma_{\mathfrak{B}}}$ associated to Brauer configuration $\Gamma_{\mathfrak{B}}$.

5.2 Multistable and Emerging Images from BCA

Let $\mathcal{R}_1, \mathcal{R}_2, \cdots, \mathcal{R}_s$, be a collection of image-repositories. Then an image $\mathcal{I}$ under the scheme of A.M. Cañadas et al. [25] is a system of the form:

$$i = (x_{i,1}, x_{i,2}, \cdots, x_{i,s}) = \sum a_{i,j} b_{i,j}, \quad a_{i,j} \in \{0,1\} \text{ where } x_{i,j} \in \mathcal{R}_j.$$

Under these circumstances, image $\mathbf{i}$ is said to be a *multistable image.*

The collection $(\mathcal{R}_1, \mathcal{R}_2, \cdots \mathcal{R}_s)$, has a matrix representation χ, such that a given image $\mathbf{i}$ has a defined fixed location in χ.

Each polygon in the Brauer configuration from $\mathbf{i}_1$ can be an *emerging image*, where

$$\begin{aligned}
\mathbf{i}_1 &= (x_{i_1,1}, x_{i_1,2}, x_{i_1,3}, \cdots, x_{i_1,s}),\\
\mathbf{i}_2 &= (x_{i_2,1}, x_{i_2,2}, x_{i_2,3}, \cdots, x_{i_2,s}),\\
\mathbf{i}_3 &= (x_{i_3,1}, x_{i_3,2}, x_{i_3,3}, \cdots, x_{i_3,s}),\\
&\vdots\\
\mathbf{i}_t &= (x_{i_t,1}, x_{i_t,2}, x_{i_t,3}, \cdots, x_{i_t,s}),
\end{aligned} \tag{5.2}$$

and where $\mathbf{i}_1$ is maximal, thus

$$\mathbf{i}_t < \cdots < \mathbf{i}_3 < \mathbf{i}_2 < \mathbf{i}_1,$$

and furthermore, $x_{i_t,j} < x_{i_{t-1},j} < x_{i_{t-2},j} < \cdots, x_{i_2,j} < x_{i_1,j}$, where $x_{i_1,j}$ is maximal and

$\mathbf{i}_t < \mathbf{i}_{t-1} < \cdots < \mathbf{i}_2 < \mathbf{i}_1$ corresponds to a maximal path in Q_Γ.

Thus, the number of images associated to the Brauer configuration is given by the formula ts^δ, where t is the number of maximal paths in Γ, s is the number of image-repositories and δ is the length of the maximal chains in Γ.

The number of images associated to the Brauer configuration permits to obtain the following important result.

Theorem 5.2.1. *The number of images associated to a scheme χ is ts^δ.*

Proof

Let $\mathcal{P}_{\Gamma_1}$ be the associated poset to the Brauer configuration $\Gamma_{\mathfrak{B}}$ (see Figure 5.9).

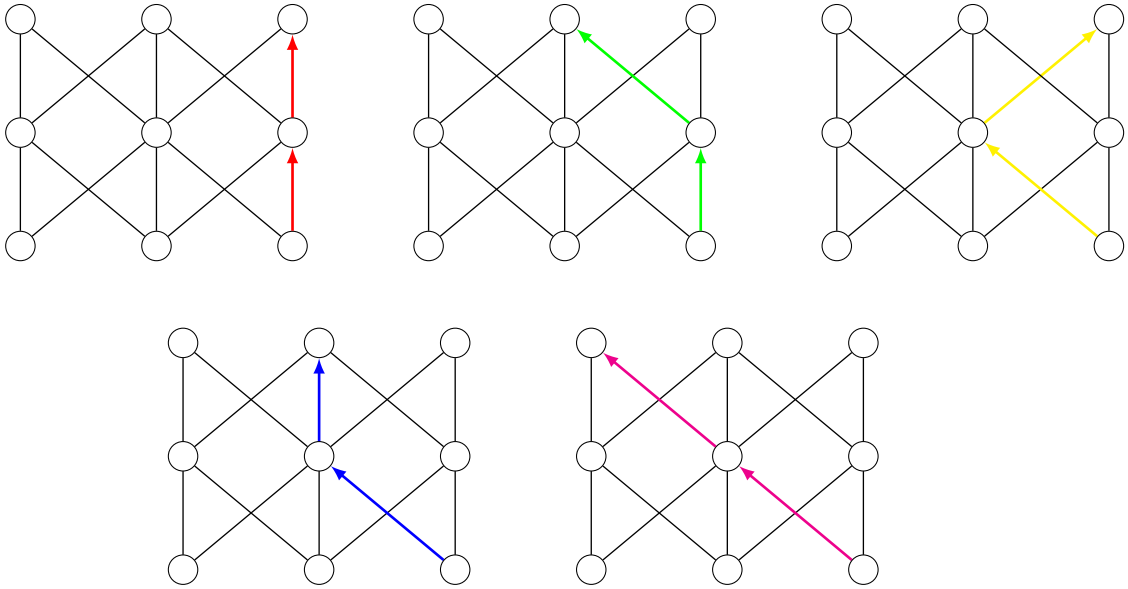


Figure 5.10: Paths $1, 2, 3, 4, 5$ from $x_{1,1}$

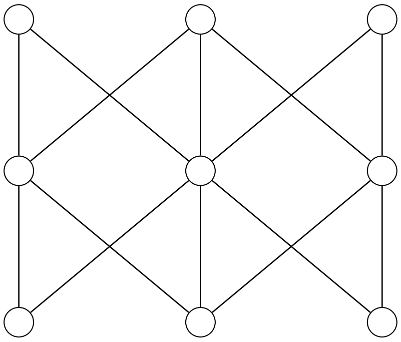


Figure 5.9: Poset $\mathcal{P}_{\Gamma_1}$ associated to $\Gamma_{\mathfrak{B}}$

From $x_{1,1}$, there are 5 paths as is shown in Figure 5.10, from $x_{2,1}$ there are 6 paths (see Figure 5.11) and from $x_{3,1}$ there are 6. The total number of paths is 16.
The paths from $x_{3,1}$ are analogous to $x_{1,1}$

Therefore, the number of images of χ associated to Poset $\mathcal{P}_1$ is $(16.5^3) = 2000$.

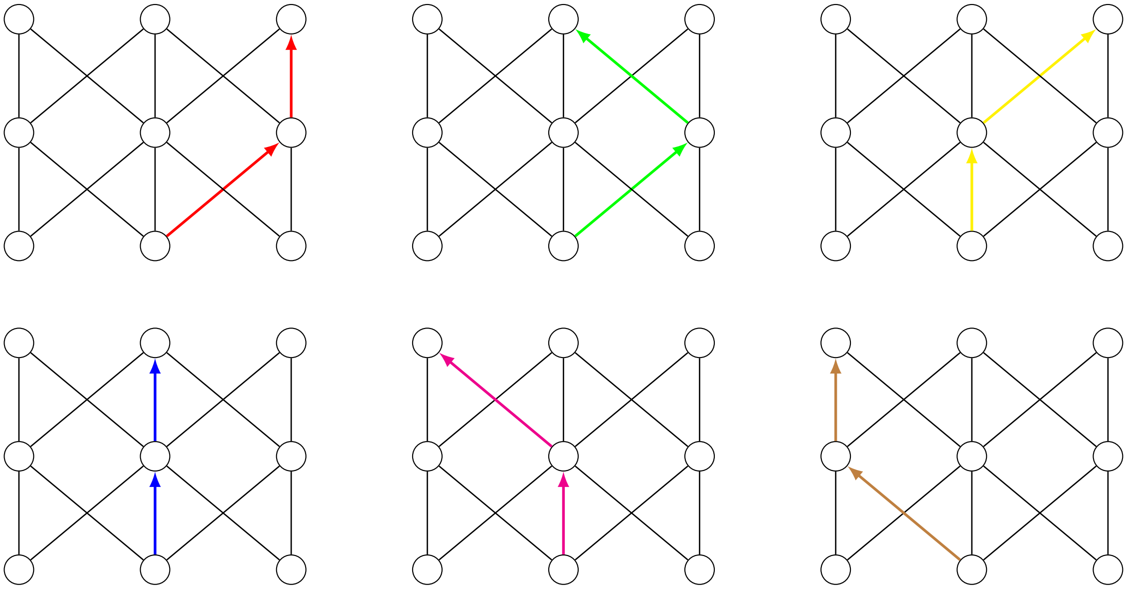


Figure 5.11: Paths $1, 2, 3, 4, 5, 6$ from $x_{2,1}$

We are done $\square$.

Chapter 6

Emerging Images from Topological Data Analysis Techniques

In this chapter, we analyze and use the tools from Persistent Homology theory in order to generate emerging images. In particular, we use topological techniques to generate space triangulations that allow us to generate random masks for a number of images-template, which can be used in HIPs (see experimental results).

6.1 Random masks construction

The construction starts with a cloud of points, then the simplicial complexes that originate a triangulation of the topological space are generated. This construction builds a random mask which is superimposed to the corresponding image in order to remove boundary data.

Let I be an image, which is a matrix of $m \times n$ values, let be $S = [1, m] \times [1, n]$. We will think of I as samples of a continuous function $f : D \longrightarrow V$.

Let $A \subset \mathbb{R}^2$ be a finite set called a point cloud, which is a sample of topological space $\mathbb{X} \subset \mathbb{R}^2$. We aim to study some topological features of $\mathbb{X}$, in such a way that only information arising from A will be used.

A sampling of the image points is taken, so that, these data constitute a point cloud. A topological Euclidean space, known as the simplicial complex is associated with the point cloud. The simplicial complex becomes an object of study. In this case the simplicial complex has been constructed as a Vietoris-Rips complex, which allows us to extract topological features from the object of study in order to obtain relevant information (see Figure 6.1).

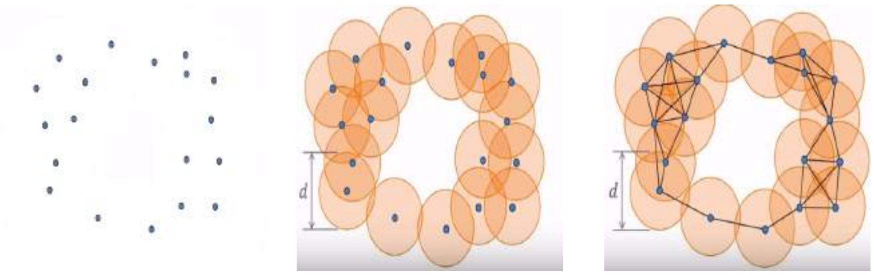


Figure 6.1: Space triangulation

A triangulation of a topological space $\mathbb{X}$ is a simplicial complex $\mathbb{K}$ along with a homeomorphism between $|\mathbb{K}|$ and $\mathbb{X}$. Henceforth, we will often blur the difference between a topological space and a simplicial complex bearing in mind that we always are handling triangulations of noisy images. New triangulations associated to the point cloud A, can be obtained by increasing the radius $\epsilon > 0$ of the different balls partitioning $\mathbb{X}$. In the Figure 6.2 is showed a triangulation which was obtained computationally with our algorithm MGA; it will be explained in next section.

Remark 6.1.1. We note that establishing the suitable radius $\epsilon > 0$ to obtain a good triangulation, constitutes an open question.

In this section, we present a practical application of space triangulation from a point cloud, in order to generate random masks whose main purpose consists of building noisy images. Actually, this procedure is an image-analogous of the ScatterType CAPTCHA introduced by Baird et al. [13]. In this application, simplicial complexes are used to build random masks such which allow vary the parameters in order to generate different models easy to recognize by humans and hard to detect by different boundary detection algorithms.

After making a triangulation for an specific radius $\epsilon > 0$, the process is stopped and the extraction of images is made. This process also allows rotations of the image to improve the perception.

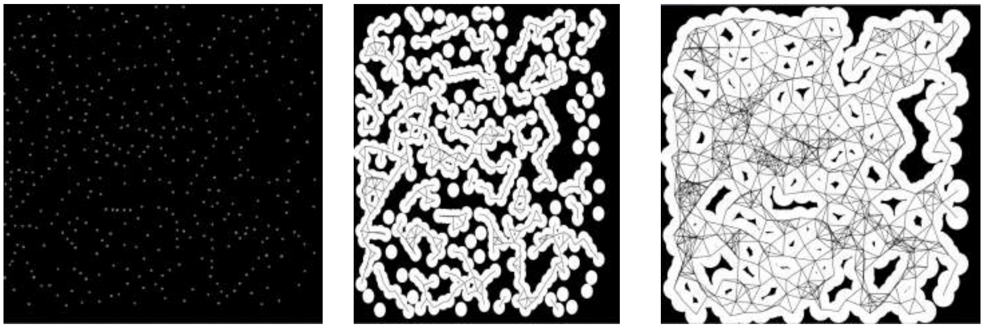


Figure 6.2: Triangulations obtained with our algorithm MGA

6.2 Masks Generating Algorithm (MGA)

In this section, we present the Masks Generating Algorithm (MGA), which allows to construct random masks from simplicial complexes.

Algorithm 6.2.1: Algorithm MGA

1. Choose the number of points p.

2. Randomly generate the point cloud with p elements.

3. Choose a radius $\epsilon > 0$ to construct balls whose center is each point.

4. If two balls intersect, connect the points (centers of the balls) with a straight line segment.

5. If three balls intersect, connect the points (centers of the balls) with a triangle.

6. Capture and record the simplicial complexes and triangulation obtained in previous steps.

7. if necessary, modify the radius ϵ of the balls to obtain a greater number of simplicial complexes and repeat steps 3 to 6.

In the following example, we present an application of how to use the space triangulation to generate an image that has been hidden. In the Figure 6.3, an sequence obtained of the image is presented. These obtained images will be used in Human Interaction Proofs (see Chapter 7).

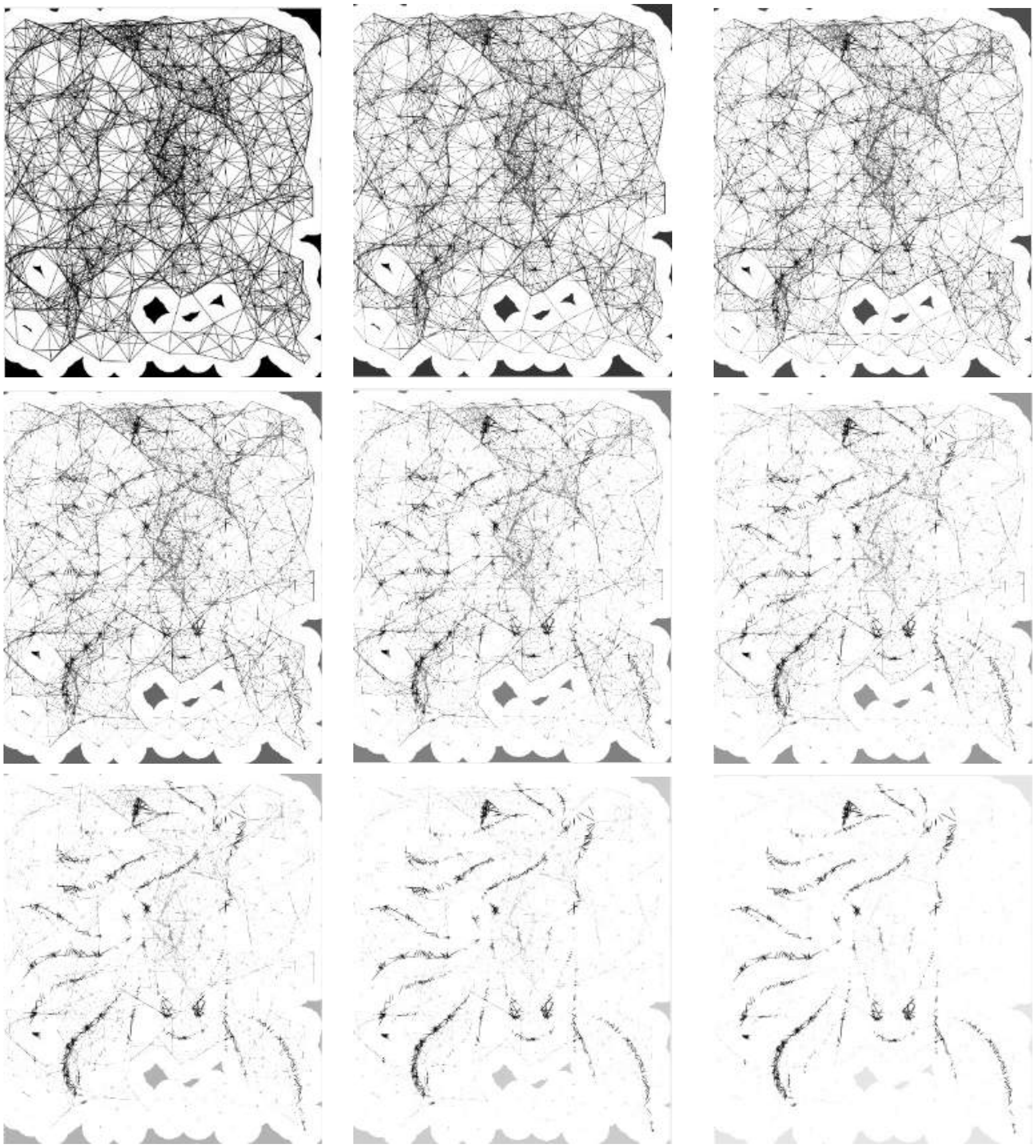


Figure 6.3: Emerging Image obtained with a random mask.

In Chapter 7 others emerging images obtained are presented, which were generated by using the exposed novel algorithm MGA.

6.3 Comparison: Homological methods and Tiled orders methods

In order to conclude this chapter, we present a comparison table about how the information is studied from homological methods and tiled orders methods to generate emerging images.

Persistent Homology	**Tiled Orders**
Points Cloud	Points Cloud
Clusters	Clusters
Construction of balls	Construction of tesellation
Construction of simplicial complex	Construction of Polygons
Homologies are studied	Posets are studied
Representation of oriented quiver Q	Representation of Poset
Intervals of Persistence and Images collection	λ-submodules

Table 6.1: Comparison: homological methods and tiled orders methods to generate emerging images.

The table shows that from two different mathematical objects, it is possible to generate collections of emerging images.

Figure 6.4 shows examples of point clouds and other space triangulations. These schemes will be used in experimental results that will be presented later. The images obtained can be seen in the appendices.

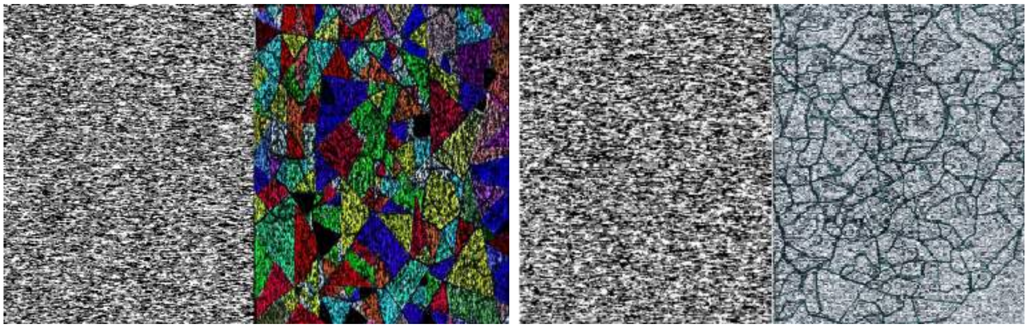


Figure 6.4: Point clouds and space triangulations.

Chapter 7

Experimental Results

In this chapter, we present the main computational results and the performed tests.

7.1 First Experiment

This first experiment uses a 01-matrix representation. A HIP named EmerCAPTCHA is applied and the results are presented in this section.

Matrix representations over $\mathbb{Z}_2$ ($\mathbb{Z}_{256}$, $\mathbb{R}$) of the poset $\mathcal{Q}$ presented in Figure 7.1 are defined to model images extracted from galleries of fairy tales, (see [88]), clowns, circus acts and the Bible books of Esther and Judith. To model these images, we have used 360000 digits of the 32th, 34th,, 35th and 38th perfect numbers, digits corresponding to Mersenne primes and of the decimals expansion of the irrational numbers $\sqrt{2}$ and $\sqrt{3}$ have been used as well. Vectors $\chi \in M_{600}$ ($\mathbb{Z}_2$) have been built and tessellated to generate models of the databases via Algorithm 3.2.1.

Several emerging and multistable images have been obtained as consequence of such construction. Experimental results with random blocks of digits in the decimal expansion of $\sqrt{2}$ and $\sqrt{3}$ allow us to conjecture that an infinite number of these models of arbitrary size can be obtained with the same databases.

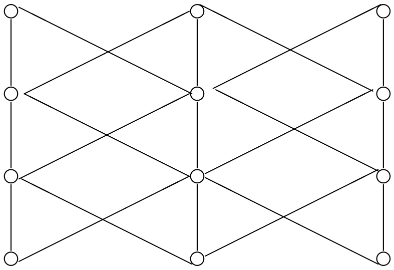
$\mathcal{Q} =$ 

Figure 7.1: Poset associated to a system of emerging and multistable images. Actually, points in this poset can be considered weak or strong according to the nature of the repositories.

These experimental results have been obtained by running *C* and MATLAB programs in a Dell Precision M4400 PC (intel core 2 Quad Extreme Edition).

In the Figure 7.2 is presented a screen with the HIP called EmerCAPTCHA, where is asked to the user the location of the eyes, mouth, nose, etc. Others images used on the HIP, also are presented.

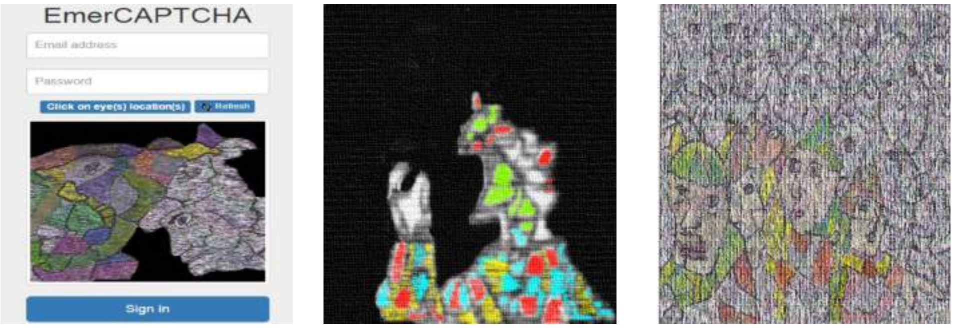
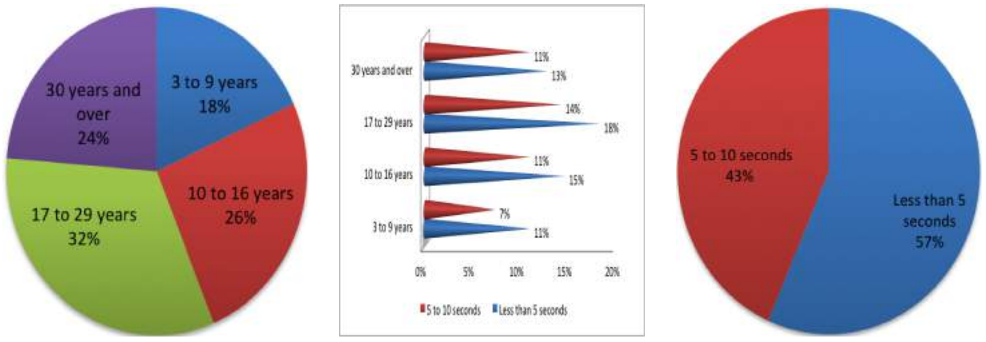


Figure 7.2: EmerCAPTCHA associated to Algorithm 3.2.1.

Different questions have been proposed to 1000 people aged between 3 to 80 years old, according to the type of CAPTCHA we have analyzed. For example, in the case of questions of type ARTiFACIAL [78] and reCAPTCHA, $4,7$ seconds was the average time to solve the test, whereas for questions of type BONGO, the average time to solve the challenge was 5.5 seconds. (See the statistical results of the applied test in Figure 7.3).



(a) (b) (c)

Figure 7.3: Figures (a)-(c) show respectively, the distribution of age, time and average time required to solve the test.

7.2 Second Experiment: Random Masks for HIPs

In this section, we present an practical application of space triangulation from a point cloud, in order to generate a random mask that converts each image of repository in a point cloud.

This procedure gives an analogous result to the images of text obtained by Baird et al. in ScatterType CAPTCHA [13].

7.2.1 Algorithm to Tell Apart Humans and Machines (ATTACH)

In this section, we present a human interaction proof based on emerging images with the main purpose of separating human beings from machines.

We use random masks, which are generated from mathematical objects and based on the Algorithm MGA, that allows us to change the parameters and generate different models.

The algorithm ATTACH describes the process for which a query associated to a masked image is presented and responded. The basic steps that should be done are showed in Algorithm 7.2.1.

Let β be a large set of images, where each image has a set of associated generic words.

Algorithm 7.2.1: Algorithm ATTACH

```
1. Randomly select $I_n$ from $\beta$.
2. Use the mask and refinement procedure to produce image $I'_n$.
3. Display $I'_n$ to user.
4. Request the user to enter the word $W$ that best describes the object he/she perceives in the display.
5. Capture, record $W$ and compare it with all the words $w_j$ in the set $L_n$.
6. If $W = w_j$ for any $j$, then "user passes the test" else, user fails the test.
```

7.2.2 Human Interactive Proof

These experimental results have been obtained by running $C++$ and MATLAB programs in a Predator 15 Acer (intel core i7, 7th generation). Figure 7.4 presents examples of the HIP generated by using the Algorithms MGA and ATTACH.

A test with different images have been presented to 500 people aged between 15 to 60 years old, all images were obtained by using the Algorithms MGA and ATTACH; $3, 8$ seconds was the average time to solve the test.

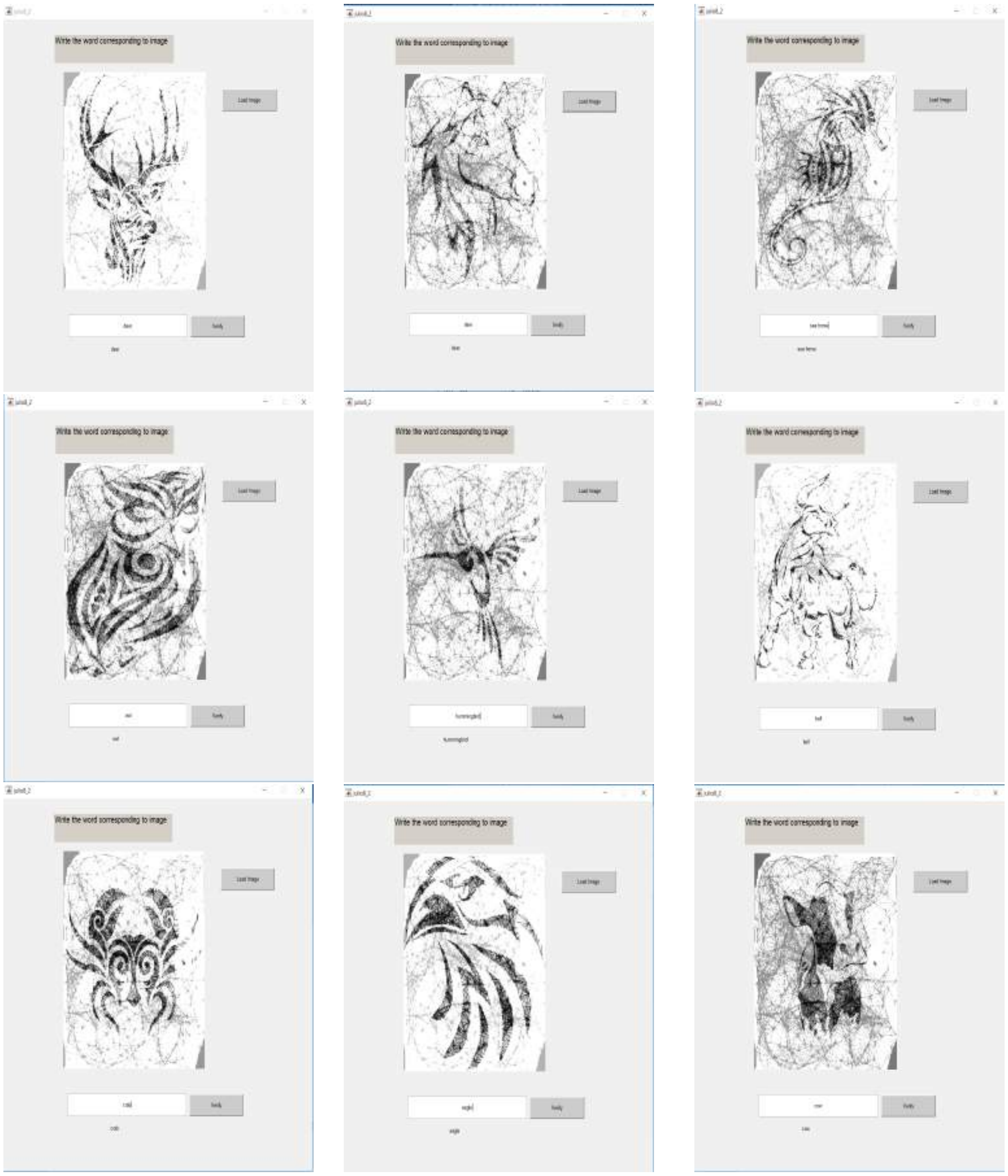


Figure 7.4: Examples: Human Interaction Proof.

7.2.3 False Positives

A false positive occurs whenever images presented to the user has loss some information about its edges or some clutter information has been added to the image. In these cases, the experience of the user is not enough to identify the model presented.

Once the experiment was conducted, 2% of the results were false positives. It corresponds to unrecognizable images for the users.

Furthermore, we realized some segmentation tests using the Prewitt and Canny algorithms. See Figure 7.5. These proofs show that the images become point clouds after the implementation of the Algorithm MGA.

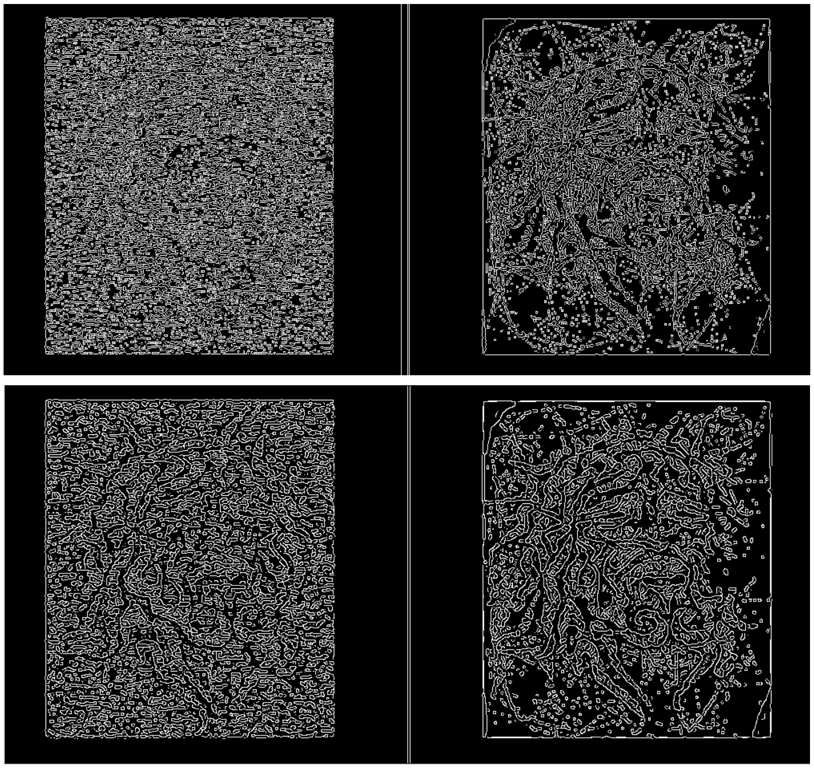


Figure 7.5: Proofs of segmentation-Prewitt and Canny.

7.3 System REIADT

System REIADT have been developed in the $C++$ program in order to generate computational results from the developed algorithms.

Figure 7.6 shows the main menu, which contains three options:

1 Sequencing.

2 Image generation.

3 Simplicial Complex.

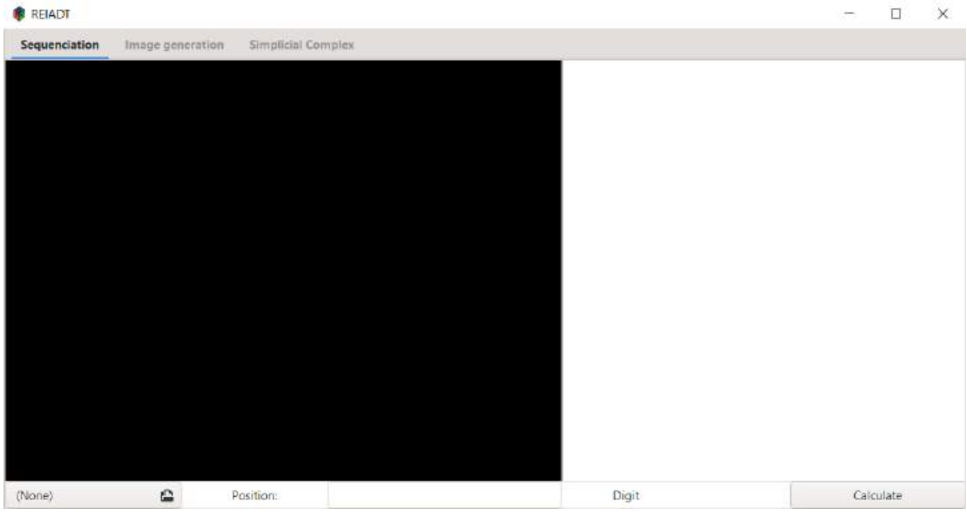


Figure 7.6: System REIADT: Main menu.

7.3.1 Sequencing Menu

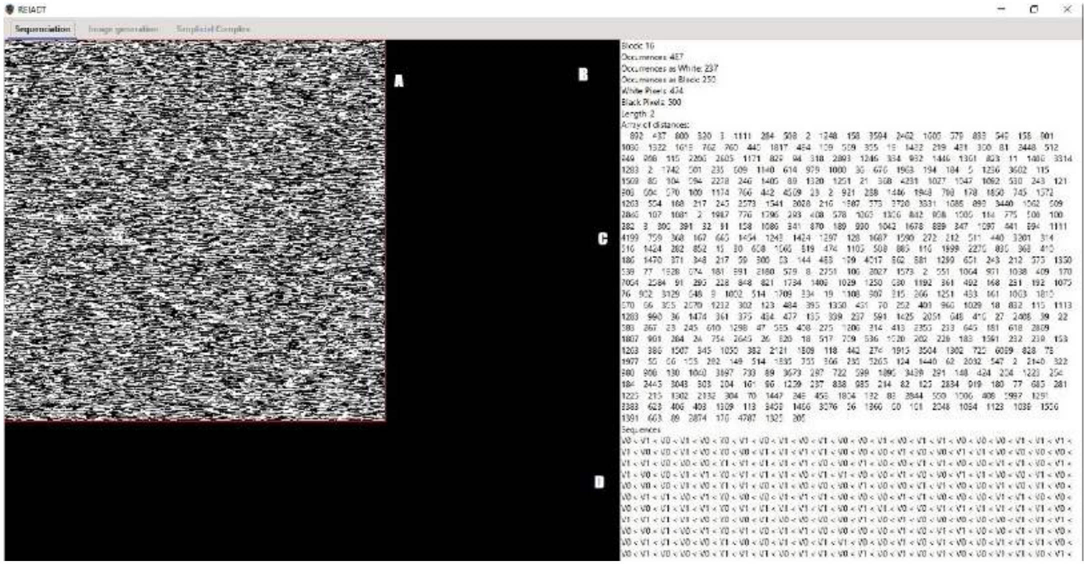
Figures 7.11 and 7.8 show screenshots with examples of outputs (sequencing process of perfect number $2^{756838}(2^{756839}-1)$). These outputs include:

A. Scheme, which shows vertices (Blocks originated by sequencing).

B. Vertex, occurrences of vertex (as white, black), length.

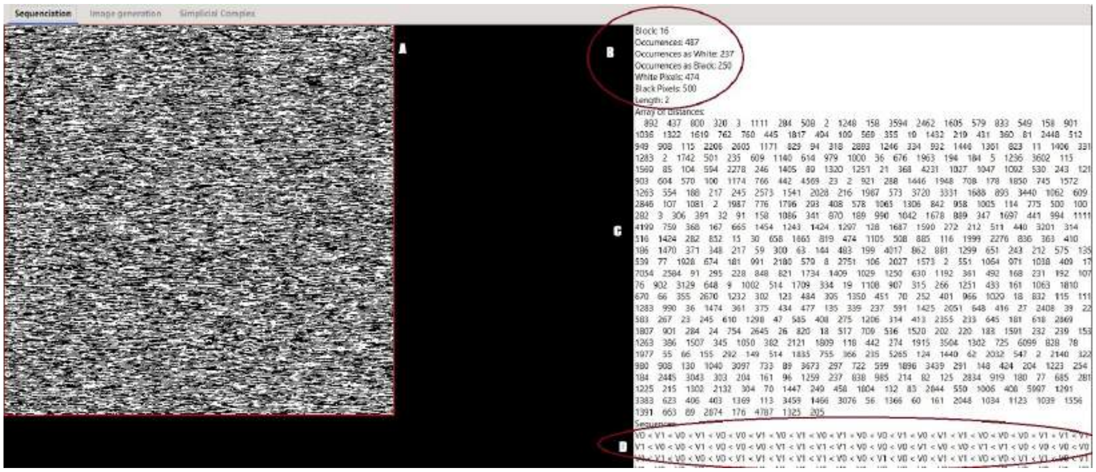
C. Array of distances.

D. Successor Sequences.



```
Block: 16
Occurrences: 487
Occurrences as White: 237
Occurrences as Black: 250
White Pixels: 474
Black Pixels: 500
Length: 2
Array of distances:
  892  437  800  320  3  1111  284  508  2  1248  158  3594  2462  1605  579  833  549  158  901
1036  1322  1619  762  760  445  1817  404  100  569  355  10  1432  219  431  360  81  2448  512
949  908  115  2206  2605  1171  829  94  318  2883  1246  334  932  1446  1361  823  11  1406  3314
1283  2  1742  501  235  609  1140  614  979  1000  36  676  1963  194  184  5  1236  3602  115
1569  85  104  594  2278  246  1405  89  1320  1251  21  368  4231  1027  1047  1092  530  243  121
903  604  570  100  1174  766  442  4569  23  2  921  288  1446  1948  708  178  1850  745  1572
1263  554  188  217  245  2573  1541  2028  216  1987  573  3720  3331  1688  893  3440  1062  609
2846  107  1081  2  1987  776  1796  293  408  578  1065  1306  842  958  1005  114  775  500  100
282  3  306  391  32  91  158  1086  341  870  189  990  1042  1678  889  347  1697  441  994  1111
4199  759  368  167  665  1454  1243  1424  1297  128  1687  1590  272  212  511  440  3201  314
516  1424  282  852  15  30  658  1665  819  474  1105  508  885  116  1999  2276  836  363  410
186  1470  371  348  217  59  300  63  144  483  199  4017  862  881  1299  651  243  212  575  1350
539  77  1928  674  181  991  2180  579  8  2751  106  2027  1573  2  551  1064  971  1038  409  170
7054  2584  91  295  228  848  821  1734  1409  1029  1250  630  1192  361  492  168  231  192  1075
76  902  3129  648  9  1002  514  1709  334  19  1108  907  315  266  1251  433  161  1063  1810
670  66  355  2670  1232  302  123  484  305  1350  451  70  252  401  966  1029  18  832  115  1113
1283  990  36  1474  361  375  434  477  135  339  237  591  1425  2051  648  416  27  2408  39  22
583  267  23  245  610  1298  47  585  408  275  1206  314  413  2355  233  645  181  618  2869
1807  901  284  24  754  2645  26  820  18  517  709  536  1520  202  220  183  1591  232  239  153
1263  386  1507  345  1050  382  2121  1809  118  442  274  1915  3504  1302  725  6099  828  78
1977  55  66  155  292  149  514  1835  755  366  235  5265  124  1440  62  2032  547  2  2140  322
980  908  130  1040  3097  733  89  3673  297  722  599  1896  3439  291  148  424  204  1223  254
184  2445  3043  303  204  161  96  1259  237  838  985  214  82  125  2834  919  180  77  685  281
1225  215  1302  2132  304  70  1447  249  458  1804  132  83  2844  550  1006  408  5997  1291
3383  623  406  403  1369  113  3459  1466  3076  56  1366  60  161  2048  1034  1123  1039  1556
1391  663  89  2874  176  4787  1325  205
Sequences:
V0 < V1 < V0 < V1 < V0 < V0 < V1 < V0 < V1 < V0 < V1 < V0 < V0 < V1 < V0 < V1 < V1 < V0 < V0 < V0 < V1 < V1 < V1 <
V1 < V0 < V0 < V1 < V0 < V0 < V0 < V1 < V1 < V1 < V1 < V0 < V0 < V0 < V1 < V0 < V0 < V1 < V1 < V0 < V0 < V0 < V0 <
V1 < V1 < V0 < V0 < V0 < V1 < V1 < V1 < V1 < V1 < V0 < V0 < V1 < V0 < V0 < V1 < V1 < V0 < V0 < V1 < V1 < V0 < V1 <
V1 < V0 < V0 < V1 < V1 < V0 < V1 < V1 < V1 < V0 < V1 < V1 < V0 < V0 < V1 < V0 < V0 < V1 < V1 < V0 < V0 < V1 < V0 <
V0 < V0 < V0 < V1 < V0 < V1 < V1 < V0 < V1 < V0 < V0 < V0 < V1 < V0 < V0 < V1 < V1 < V0 < V0 < V1 < V0 < V1 < V0 <
V0 < V1 < V1 < V0 < V1 < V0 < V0 < V0 < V1 < V1 < V1 < V0 < V0 < V0 < V1 < V1 < V0 < V0 < V0 < V1 < V1 < V0 < V0 <
V0 < V0 < V1 < V1 < V1 < V0 < V1 < V1 < V1 < V1 < V1 < V0 < V1 < V1 < V0 < V0 < V1 < V1 < V1 < V1 < V1 < V1 < V0 <
V1 < V1 < V1 < V1 < V0 < V0 < V0 < V1 < V0 < V0 < V0 < V0 < V1 < V0 < V1 < V0 < V0 < V0 < V0 < V0 < V1 < V1 < V1 <
V0 < V0 < V1 < V0 < V1 < V0 < V1 < V0 < V1 < V1 < V0 < V0 < V0 < V0 < V1 < V0 < V1 < V1 < V0 < V0 < V1 < V1 < V0 <
V0 < V1 < V1 < V0 < V0 < V1 < V0 < V0 < V1 < V0 < V1 < V1 < V0 < V0 < V1 < V0 < V1 < V0 < V0 < V1 < V1 < V1 < V0 <
V0 < V1 < V0 < V0 < V0 < V1 < V1 < V1 < V0 < V1 < V1 < V0 < V1 < V0 < V1 < V0 < V1 < V0 < V0 < V1 < V1 < V0 < V1 <
```

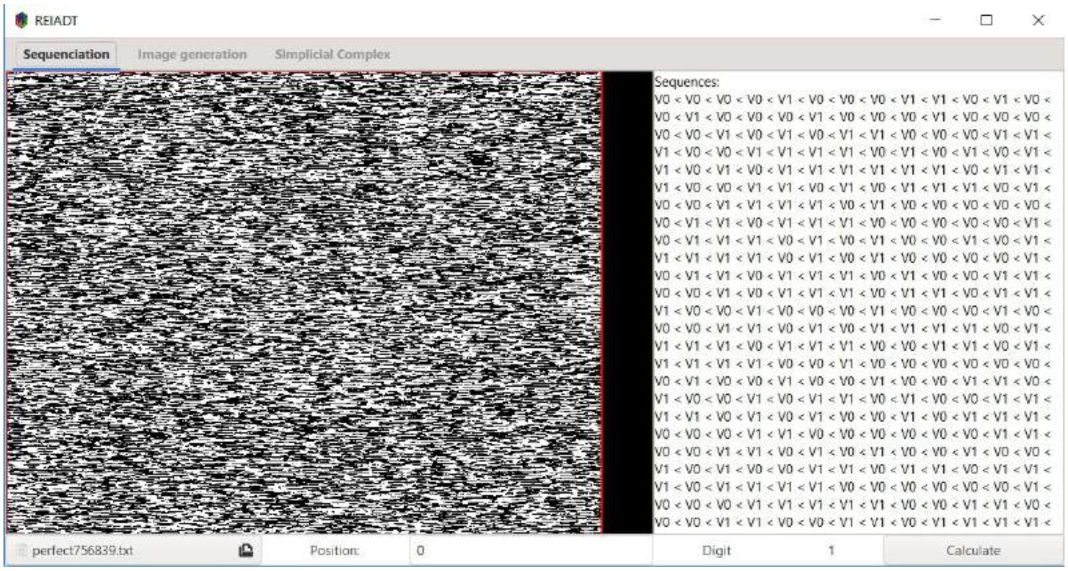
Figure 7.7: Output: sequencing process.



```
Block: 16
Occurrences: 487
Occurrences as White: 237
Occurrences as Black: 250
White Pixels: 474
Black Pixels: 500
Length: 2
Array of distances:
  892  437  800  320  3  1111  284  508  2  1248  158  3594  2462  1605  579  833  549  158  901
1036  1322  1619  762  760  445  1817  404  100  569  355  10  1432  219  431  360  81  2448  512
949  908  115  2206  2605  1171  829  94  318  2883  1246  334  932  1446  1361  823  11  1406  331
1283  2  1742  501  235  609  1140  614  979  1000  36  676  1963  194  184  5  1236  3602  115
1569  85  104  594  2278  246  1405  89  1320  1251  21  368  4231  1027  1047  1092  530  243  121
903  604  570  100  1174  766  442  4569  23  2  921  288  1446  1948  708  178  1850  745  1572
1263  554  188  217  245  2573  1541  2028  216  1987  573  3720  3331  1688  893  3440  1062  609
2846  107  1081  2  1987  776  1796  293  408  578  1065  1306  842  958  1005  114  775  500  100
282  3  306  391  32  91  158  1086  341  870  189  990  1042  1678  889  347  1697  441  994  1111
4199  759  368  167  665  1454  1243  1424  1297  128  1687  1590  272  212  511  440  3201  314
516  1424  282  852  15  30  658  1665  819  474  1105  508  885  116  1999  2276  836  363  410
186  1470  371  348  217  59  300  63  144  483  199  4017  862  881  1299  651  243  212  575  135
539  77  1928  674  181  991  2180  579  8  2751  106  2027  1573  2  551  1064  971  1038  409  17
7054  2584  91  295  228  848  821  1734  1409  1029  1250  630  1192  361  492  168  231  192  107
76  902  3129  648  9  1002  514  1709  334  19  1108  907  315  266  1251  433  161  1063  1810
670  66  355  2670  1232  302  123  484  305  1350  451  70  252  401  966  1029  18  832  115  111
1283  990  36  1474  361  375  434  477  135  339  237  591  1425  2051  648  416  27  2408  39  22
583  267  23  245  610  1298  47  585  408  275  1206  314  413  2355  233  645  181  618  2869
1807  901  284  24  754  2645  26  820  18  517  709  536  1520  202  220  183  1591  232  239  153
1263  386  1507  345  1050  382  2121  1809  118  442  274  1915  3504  1302  725  6099  828  78
1977  55  66  155  292  149  514  1835  755  366  235  5265  124  1440  62  2032  547  2  2140  322
980  908  130  1040  3097  733  89  3673  297  722  599  1896  3439  291  148  424  204  1223  254
184  2445  3043  303  204  161  96  1259  237  838  985  214  82  125  2834  919  180  77  685  281
1225  215  1302  2132  304  70  1447  249  458  1804  132  83  2844  550  1006  408  5997  1291
3383  623  406  403  1369  113  3459  1466  3076  56  1366  60  161  2048  1034  1123  1039  1556
1391  663  89  2874  176  4787  1325  205
Sequences:
V0 < V1 < V0 < V1 < V0 < V0 < V1 < V0 < V1 < V0 < V1 < V0 < V0 < V1 < V0 < V1 < V1 < V0 < V0 < V0 < V1 < V1 < V1
V1 < V0 < V0 < V1 < V0 < V0 < V0 < V1 < V1 < V1 < V1 < V0 < V0 < V0 < V1 < V0 < V0 < V1 < V1 < V0 < V0 < V0 < V0
V1 < V1 < V0 < V0 < V0 < V1 < V1 < V1 < V1 < V1 < V0 < V0 < V1 < V0 < V0 < V1 < V1 < V0 < V0 < V1 < V1 < V0 < V1
```

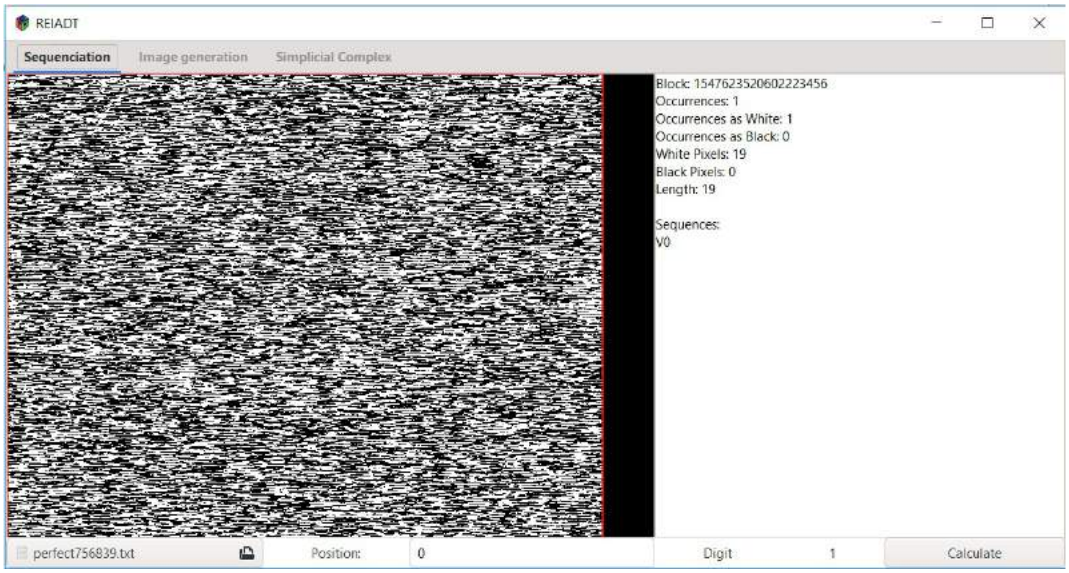
Figure 7.8: Computational process of sequencing.

In Figures 7.9 and 7.10 the successor sequences of different vertices are showed.



```
Sequences:
V0 < V0 < V0 < V0 < V1 < V0 < V0 < V0 < V1 < V1 < V0 < V1 < V0 <
V0 < V1 < V0 < V0 < V0 < V1 < V0 < V0 < V0 < V1 < V0 < V0 < V0 <
V0 < V0 < V1 < V0 < V1 < V0 < V1 < V1 < V0 < V0 < V0 < V1 < V1 <
V1 < V0 < V0 < V1 < V1 < V1 < V1 < V0 < V1 < V0 < V1 < V0 < V1 <
V1 < V0 < V1 < V0 < V1 < V1 < V1 < V1 < V1 < V1 < V0 < V1 < V1 <
V1 < V0 < V0 < V1 < V1 < V0 < V1 < V0 < V1 < V1 < V1 < V0 < V1 <
V0 < V0 < V1 < V1 < V1 < V1 < V0 < V1 < V0 < V0 < V0 < V0 < V0 <
V0 < V1 < V1 < V0 < V1 < V1 < V1 < V0 < V0 < V0 < V0 < V0 < V1 <
V0 < V1 < V1 < V1 < V0 < V1 < V0 < V1 < V0 < V0 < V1 < V0 < V1 <
V1 < V1 < V1 < V1 < V0 < V1 < V0 < V1 < V0 < V0 < V0 < V0 < V1 <
V0 < V1 < V1 < V0 < V1 < V1 < V1 < V0 < V1 < V0 < V0 < V1 < V1 <
V0 < V0 < V1 < V0 < V1 < V1 < V1 < V0 < V1 < V1 < V0 < V1 < V1 <
V1 < V0 < V0 < V1 < V0 < V0 < V0 < V1 < V0 < V0 < V0 < V1 < V0 <
V0 < V0 < V1 < V1 < V0 < V1 < V0 < V1 < V1 < V1 < V1 < V0 < V1 <
V1 < V1 < V1 < V0 < V1 < V1 < V1 < V0 < V0 < V1 < V1 < V0 < V1 <
V1 < V1 < V1 < V1 < V0 < V0 < V1 < V0 < V0 < V0 < V0 < V0 < V0 <
V0 < V1 < V0 < V0 < V1 < V0 < V0 < V1 < V0 < V0 < V1 < V1 < V0 <
V1 < V0 < V0 < V1 < V0 < V1 < V1 < V0 < V1 < V0 < V0 < V1 < V1 <
V1 < V1 < V0 < V1 < V0 < V1 < V0 < V0 < V1 < V0 < V1 < V1 < V1 <
V0 < V0 < V0 < V1 < V1 < V0 < V0 < V0 < V0 < V0 < V0 < V1 < V1 <
V0 < V0 < V1 < V1 < V0 < V1 < V0 < V1 < V0 < V0 < V1 < V0 < V0 <
V1 < V0 < V1 < V0 < V0 < V1 < V1 < V0 < V1 < V1 < V0 < V1 < V1 <
V1 < V0 < V1 < V1 < V1 < V1 < V0 < V0 < V0 < V0 < V0 < V0 < V1 <
V0 < V0 < V0 < V1 < V1 < V1 < V1 < V1 < V0 < V0 < V1 < V1 < V0 <
V0 < V0 < V1 < V1 < V0 < V0 < V1 < V1 < V0 < V1 < V1 < V1 < V1 <
```

Figure 7.9: Output: sequencing process.

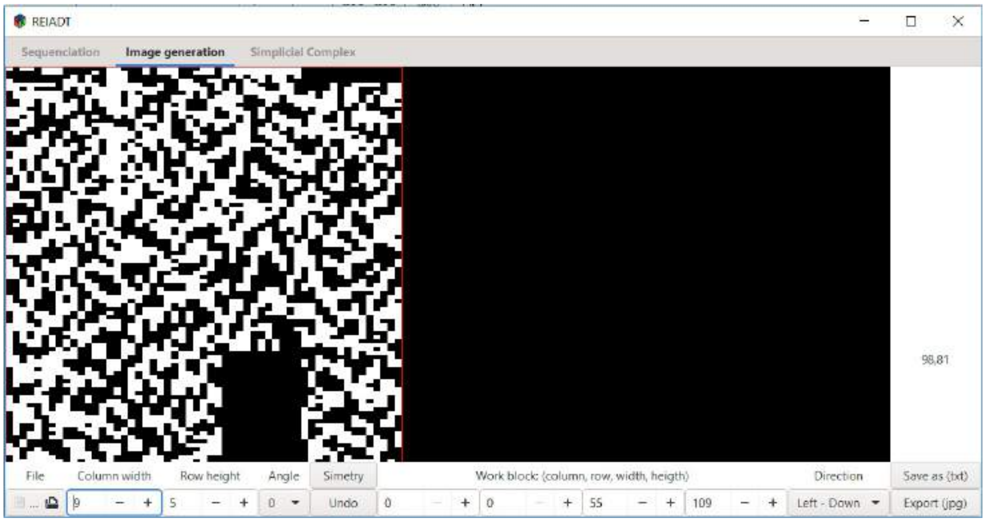


```
Block: 1547623520602223456
Occurrences: 1
Occurrences as White: 1
Occurrences as Black: 0
White Pixels: 19
Black Pixels: 0
Length: 19

Sequences:
V0
```

Figure 7.10: Computational process of sequencing (truncated vertex) .

7.3.2 Image Generation Menu



Column width 9

Row height 5

Angle 0

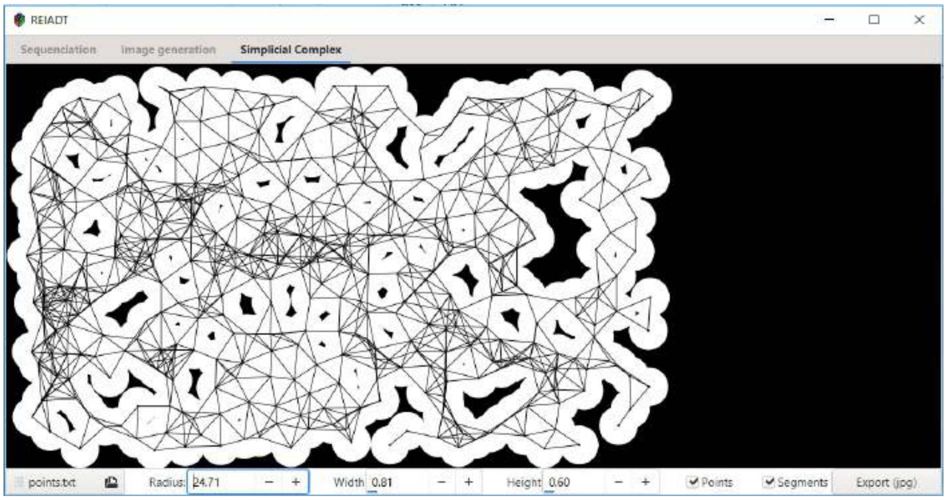
Work block: (column, row, width, heigth) 0 0 55 109

Direction Left - Down

Figure 7.11: Computational process of image generation.

7.3.3 Simplicial Complex Menu

This process allows us to generate random mask from point clouds (see Figures 7.12 and 7.13).



points.txt

Radius: 24.71

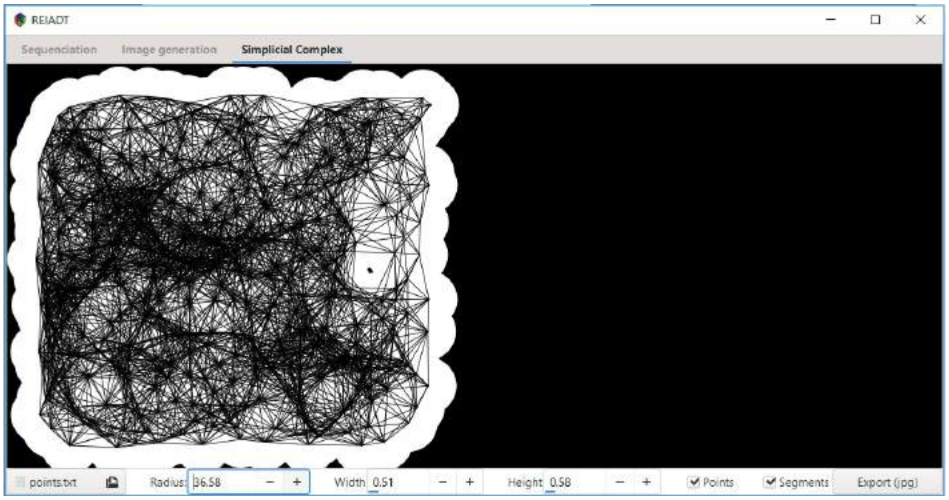
Width 0.81

Height 0.60

☑ Points

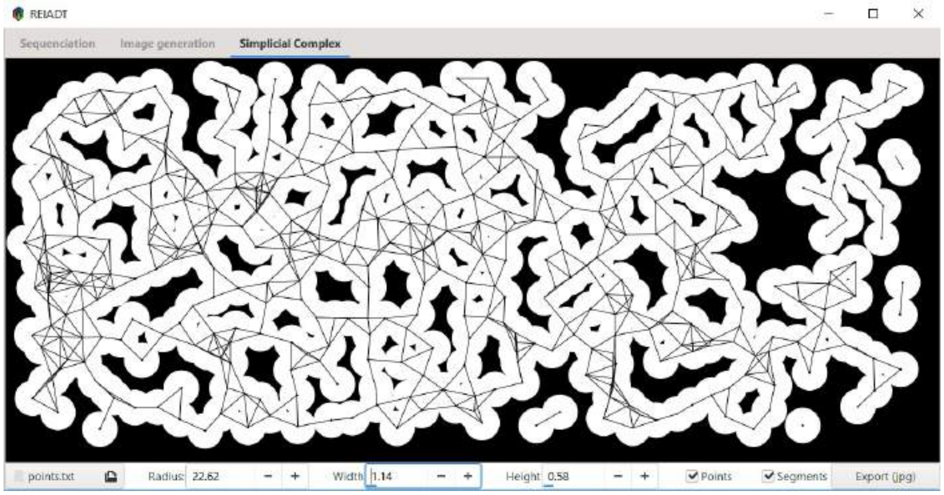
☑ Segments

Figure 7.12: Computational process of a simplicial complex.



Radius: 36.58 Width 0.51 Height 0.58 ☑ Points ☑ Segments

Figure 7.13: Computational process of a simplicial complex.



Radius: 22.62 Width 1.14 Height 0.58 ☑ Points ☑ Segments

Figure 7.14: Computational process of a simplicial complex.

Chapter 8

Conclusions and Future work

Matrix problems and in particular matrix representations of partially ordered sets (posets) are used to formally define and generate emerging and multistable images. Images induced by such representations are mosaics which can be used to design different types of Human Interaction Proofs.

Semimaximal rings known as tiled orders and some Brauer configurations are mathematical structures that can be used to encode and decode emerging image-repositories.

01-Tiled orders are used to interpret images as Posets. Associating an image to the structure of poset allows us to generate a new filter to eliminate noise-image . This filter is groundbreaking, because filters that exists these days are based on statistical methods and other mathematical concepts.

Representations of posets induced by digits of large positive integers can be processed in order to obtain mosaics, emerging and multistable images from databases of fairy tales, circus acts and the Bible's texts.

Due that solving Human Interactive proofs (HIPs) requires greater involvement of the human visual system, we can conclude that different types of Human Interaction Proofs can be enhanced, provided that the corresponding questions concern emerging images and multistable images.

Using Brauer configurations to study the information (when it is crossed), is not just better than using concepts of persistence Homology, but allows us to explore more the images processing by mean of this technique.

In particular, persistence Homology performs multi-scale analysis on a data set and allows to identify clusters, holes and voids therein. These latter topological structures complement standard feature extractor of the data generating new open problems in Artificial Intelligence.

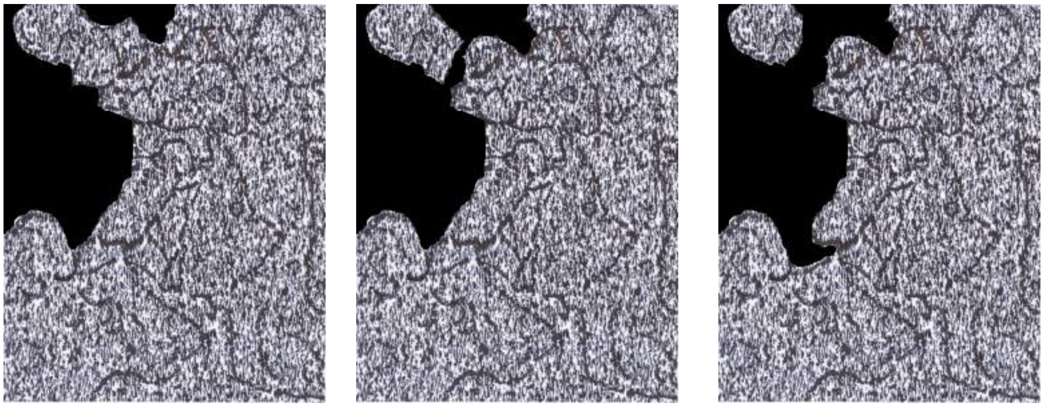
Since coherent information can be extracted from digits in the decimal expansion of the irrational number $\sqrt{3}$, we can conjecture a positive answer to the Leibniz's question in this case and that digits in the decimal expansion of some irrational numbers generate an infinite number of emerging images, which is currently one of the major concerns when using this type of images to design HIPs.

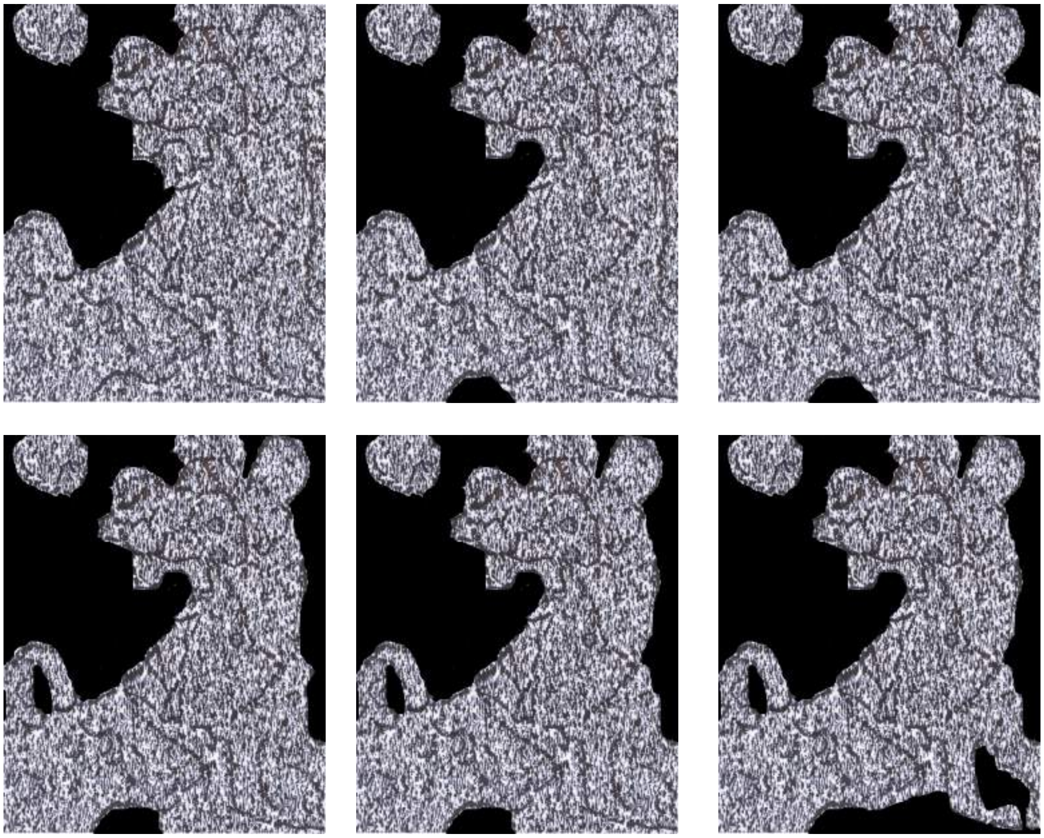
A complete classification of irrational numbers, Mersenne primes and perfect numbers according to the graphical information provided by these numbers must be done in the future.

Appendix A

Examples of Extraction of images

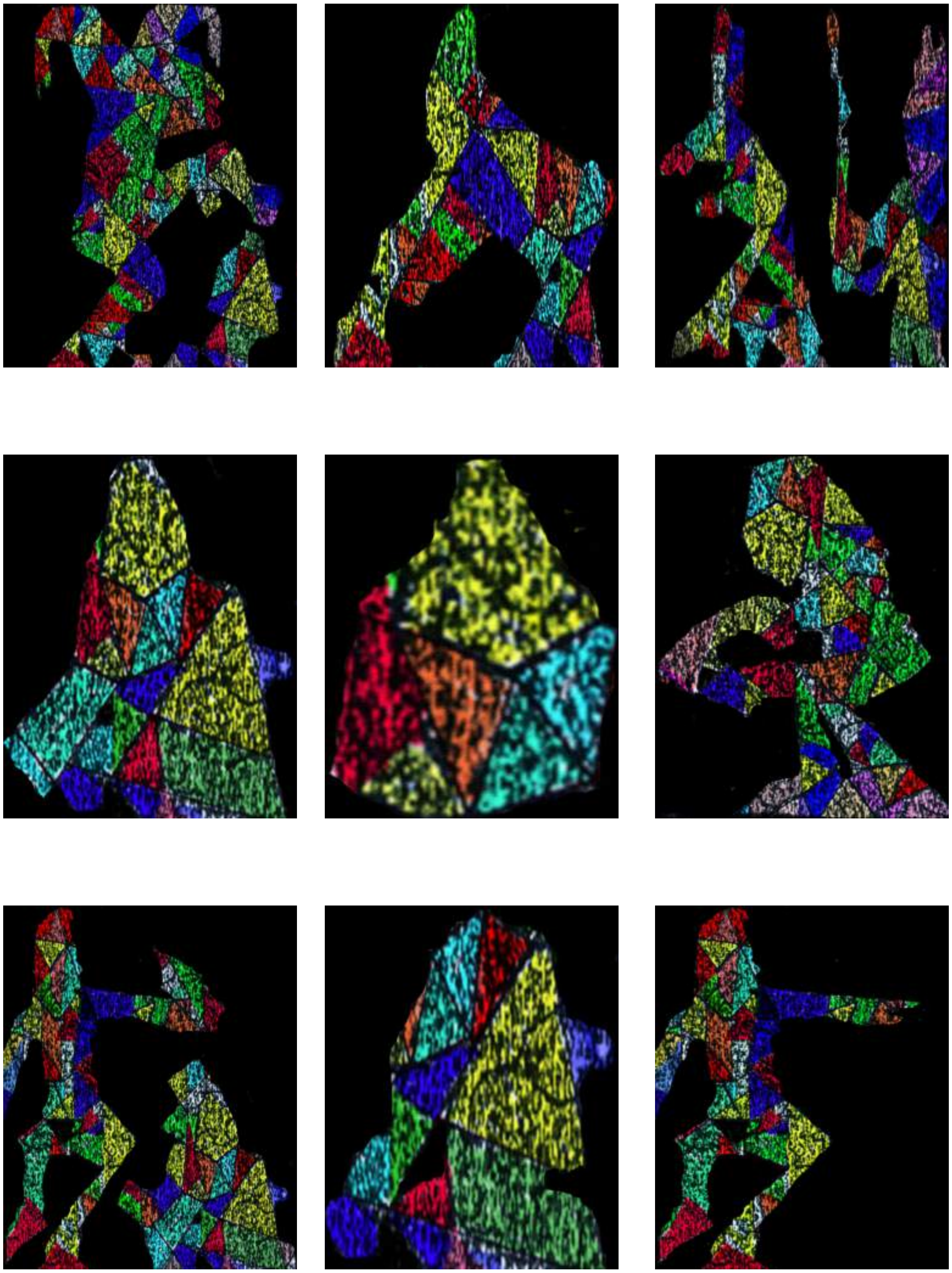
In this appendix, we show some experimental results obtained by applying the Algorithms 3.2.1 and 4.2.2.

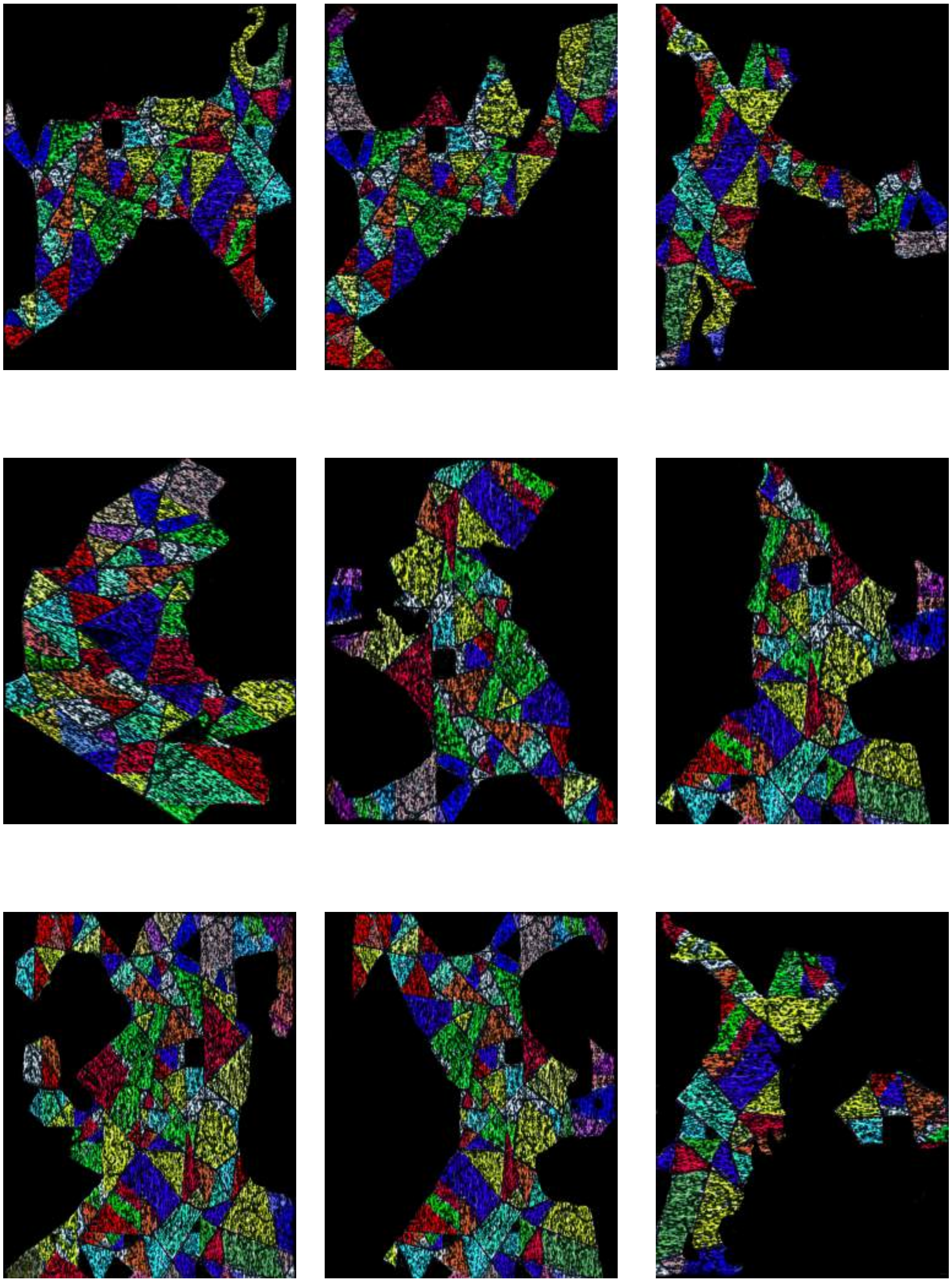


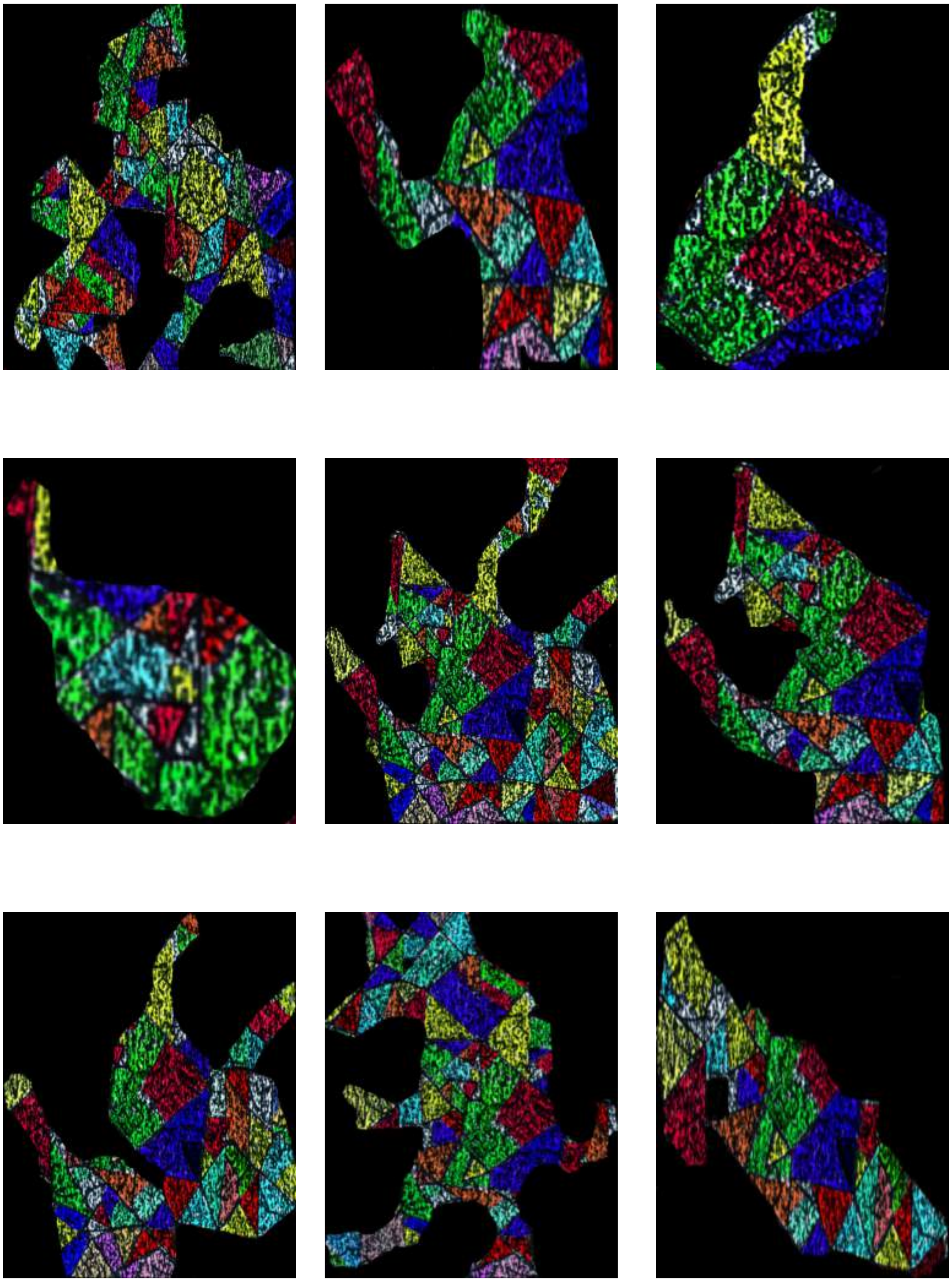


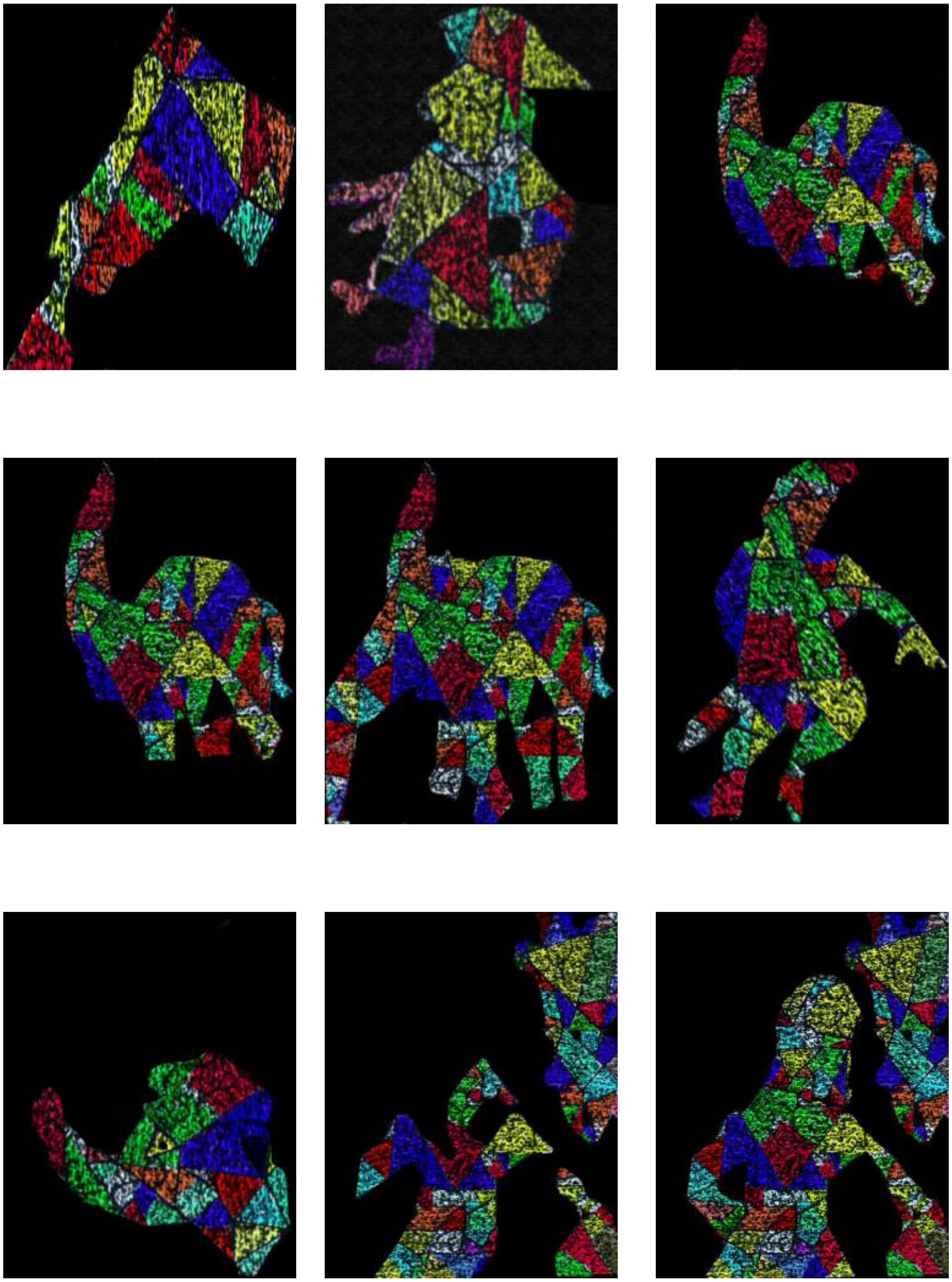
Examples of Mosaic images

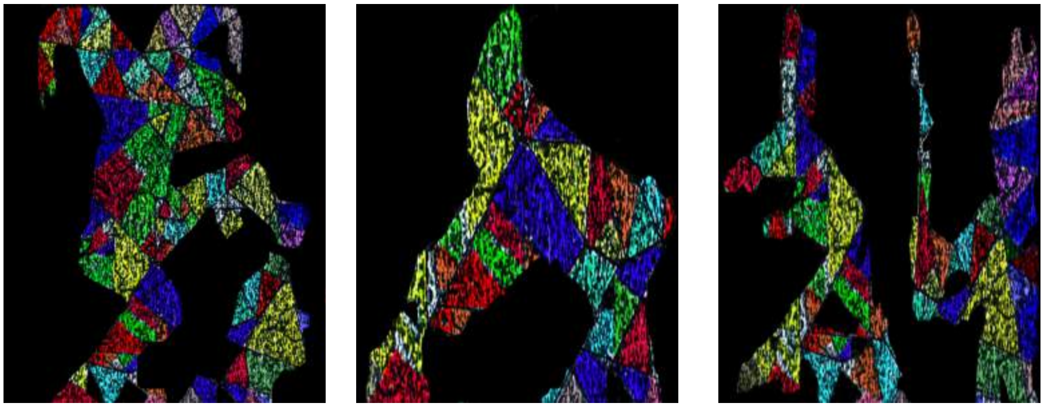
The following images were obtained from Cañadas's Scheme by using Brauer configurations.





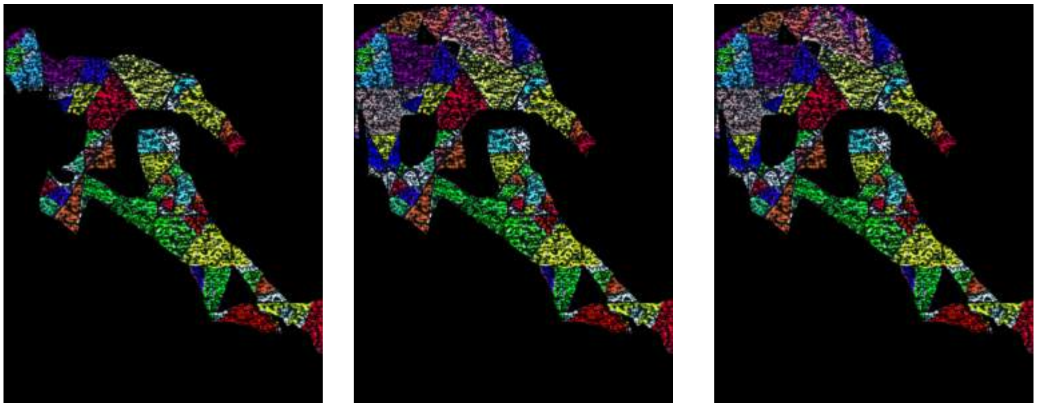


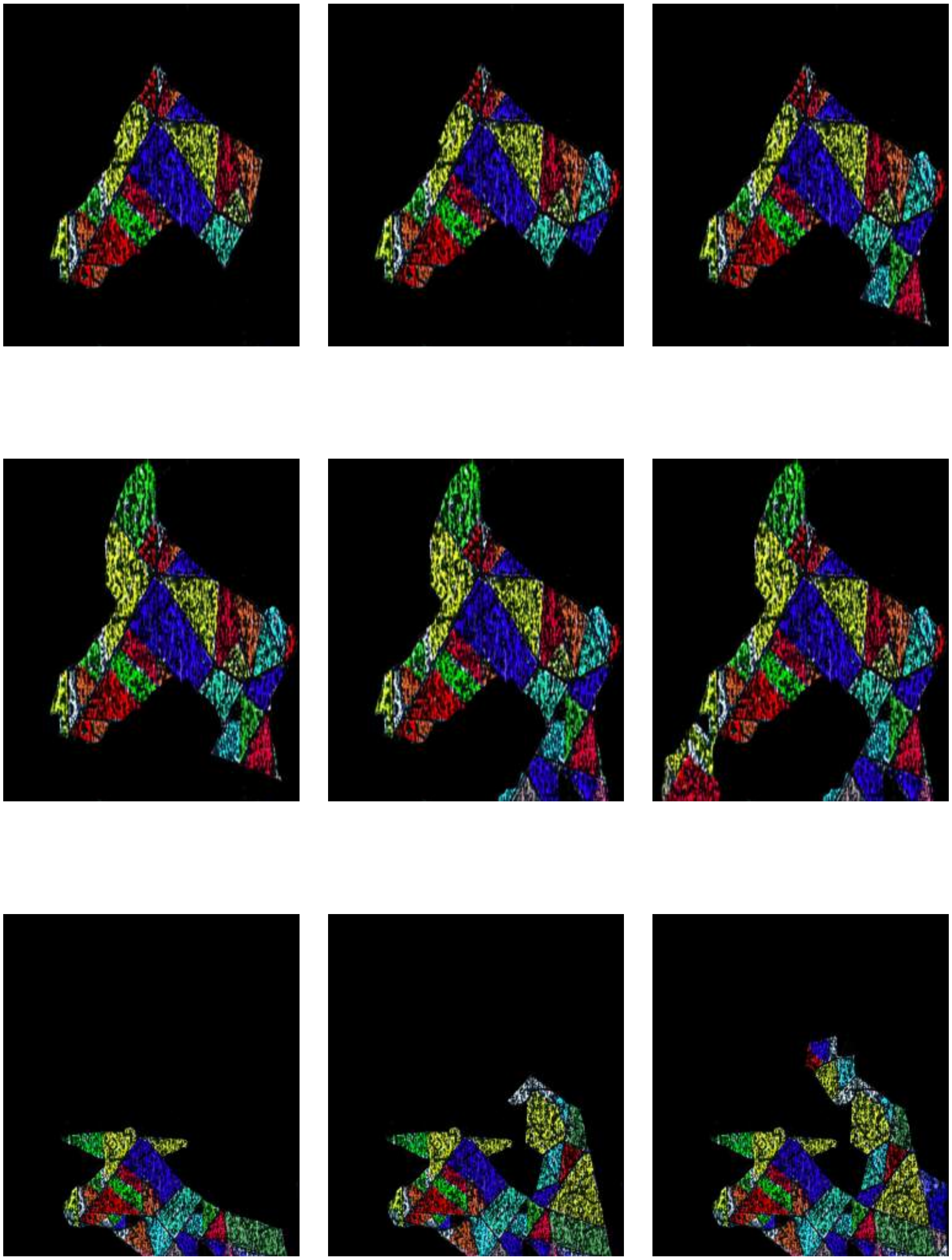


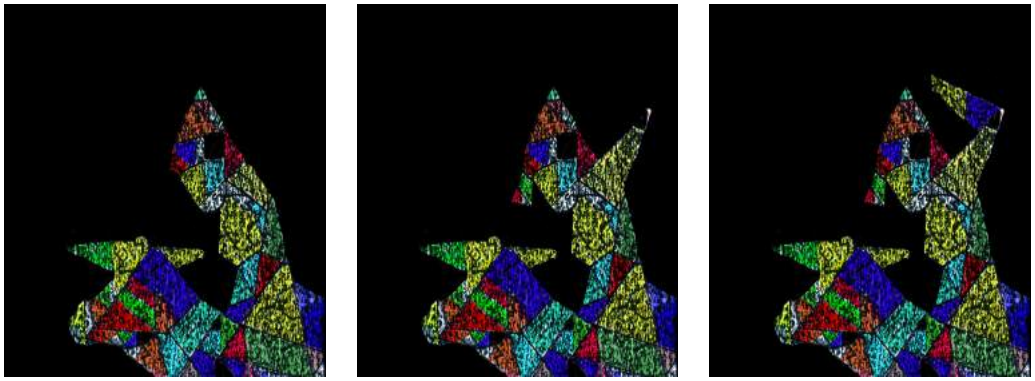


Appendix B

Examples of Sequences of Polygons

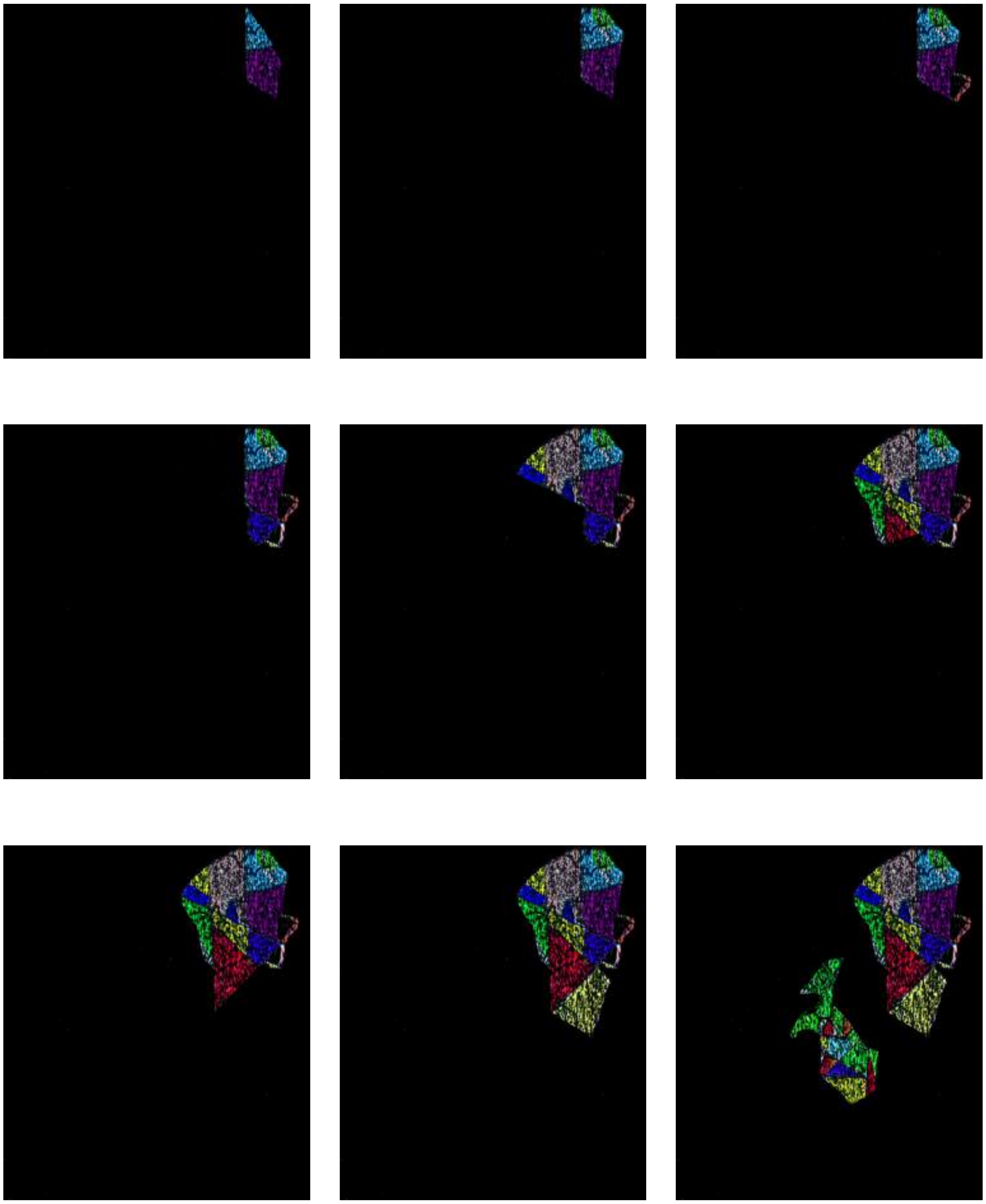


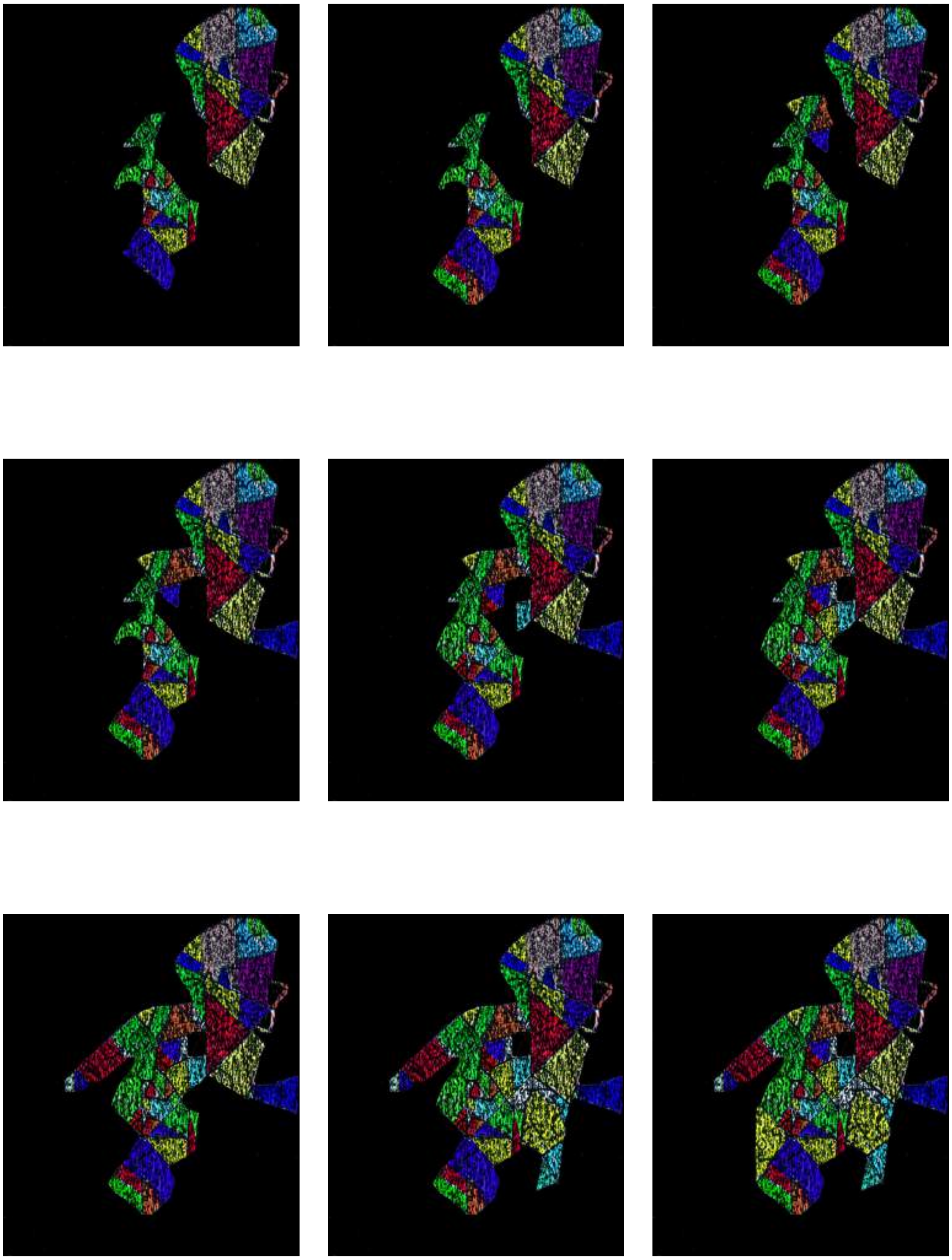


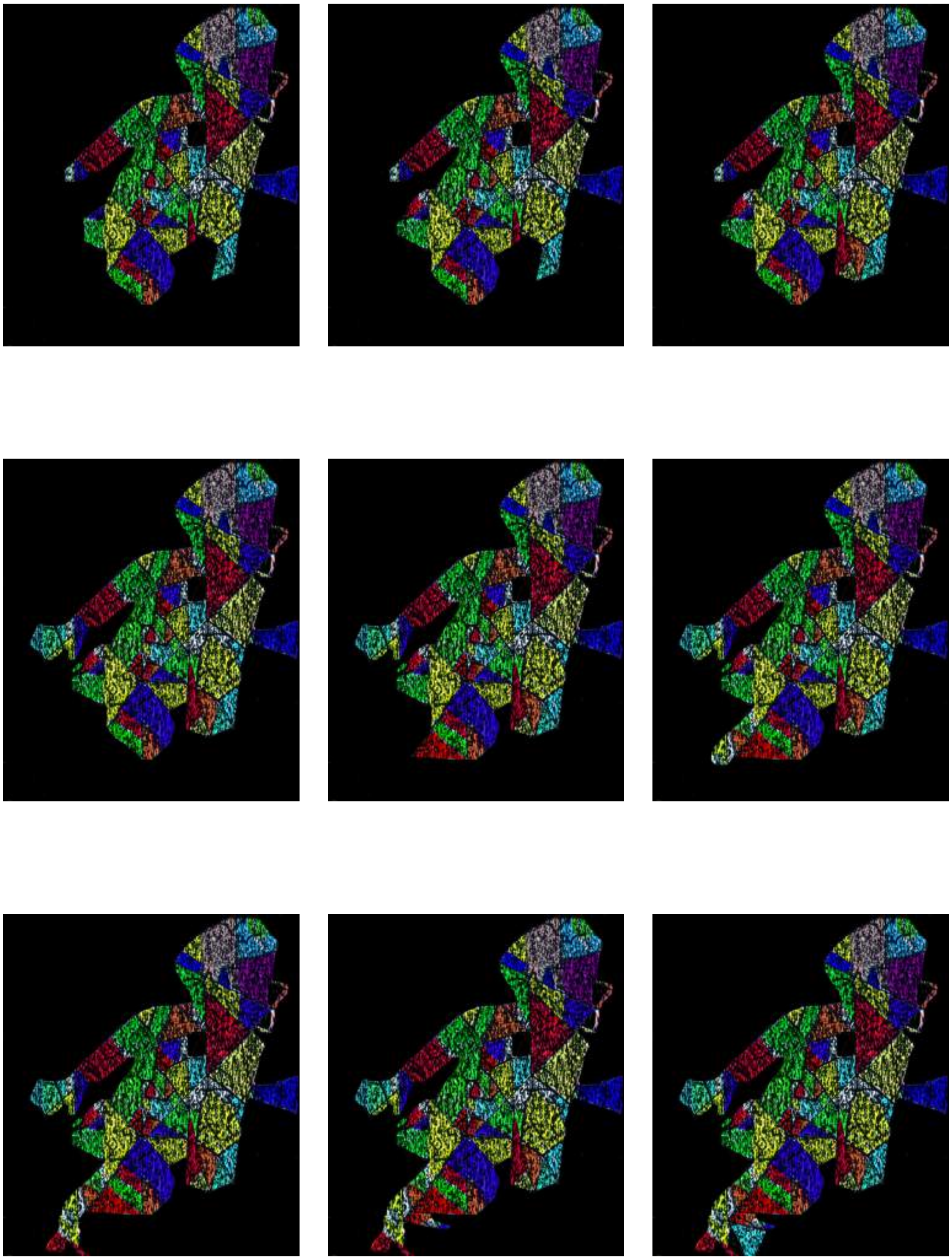


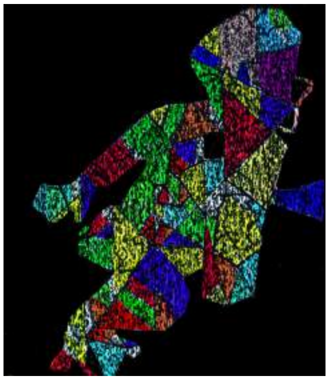
B.1 Sequences of Images

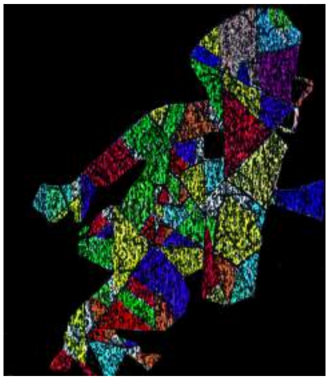
In the following examples the images sequences are presented.

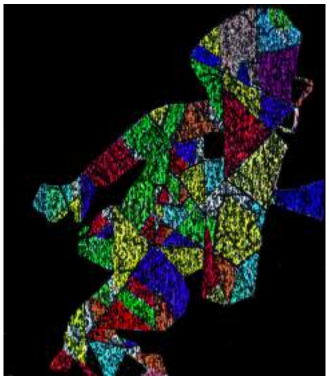


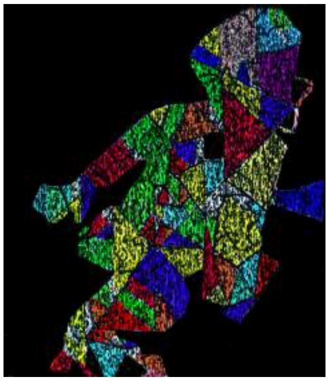


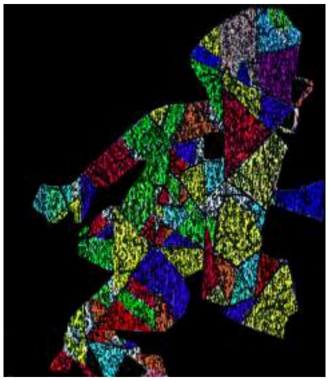


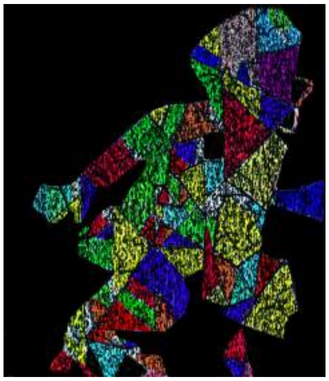


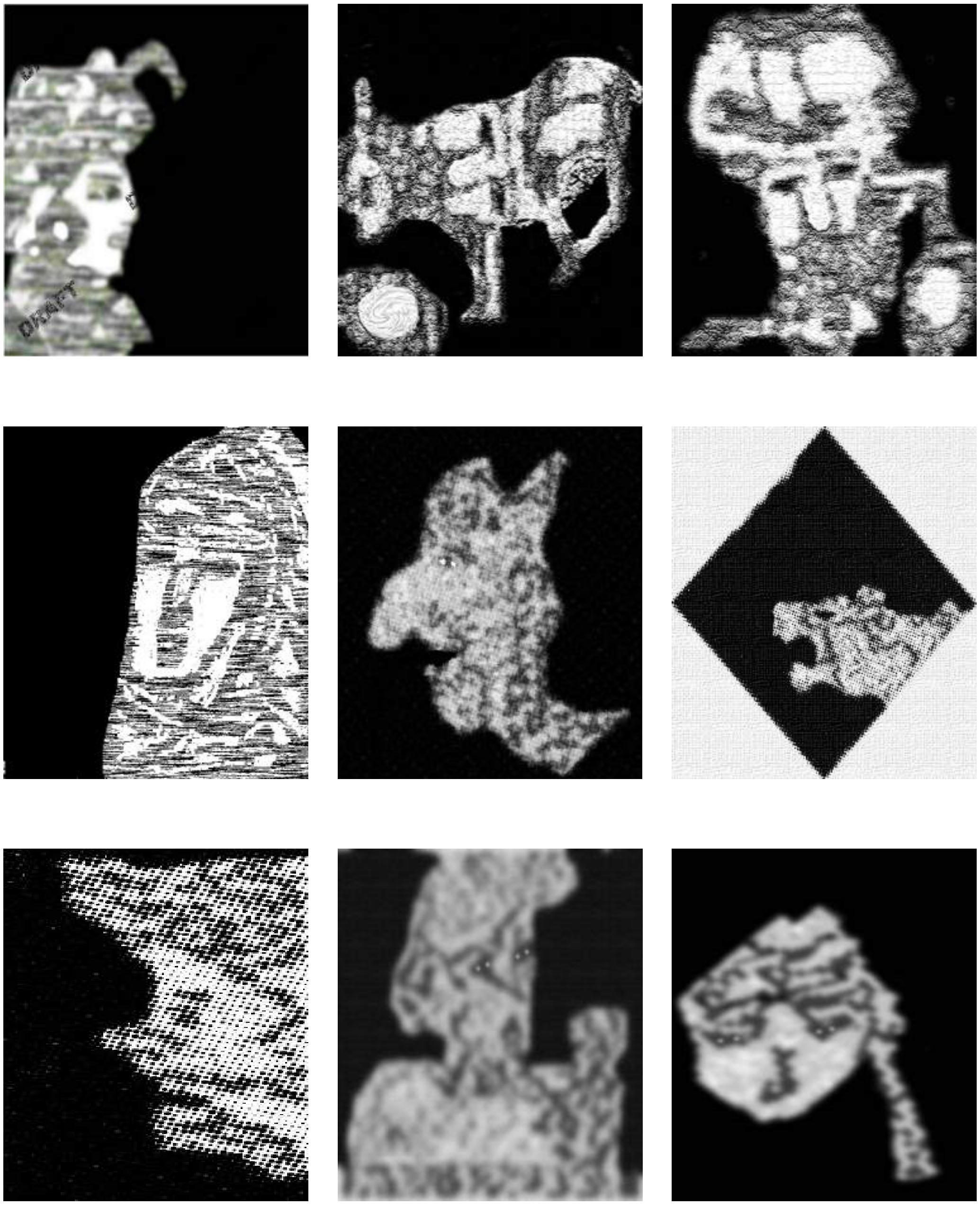


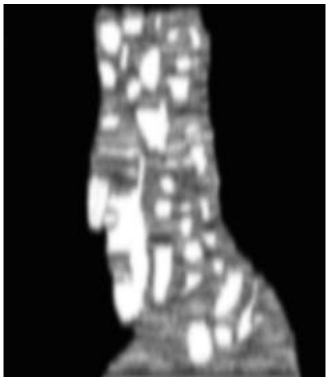


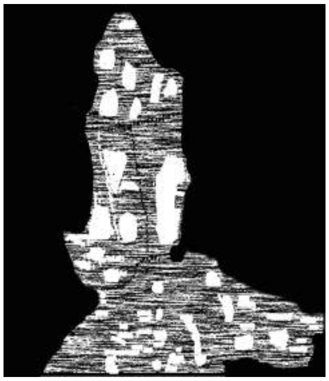


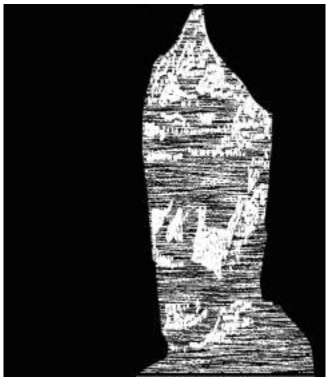
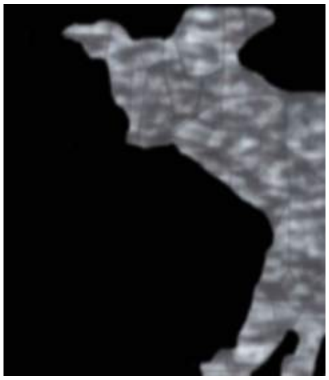


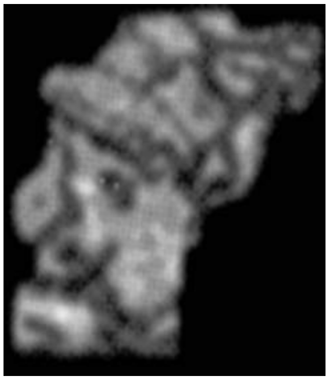


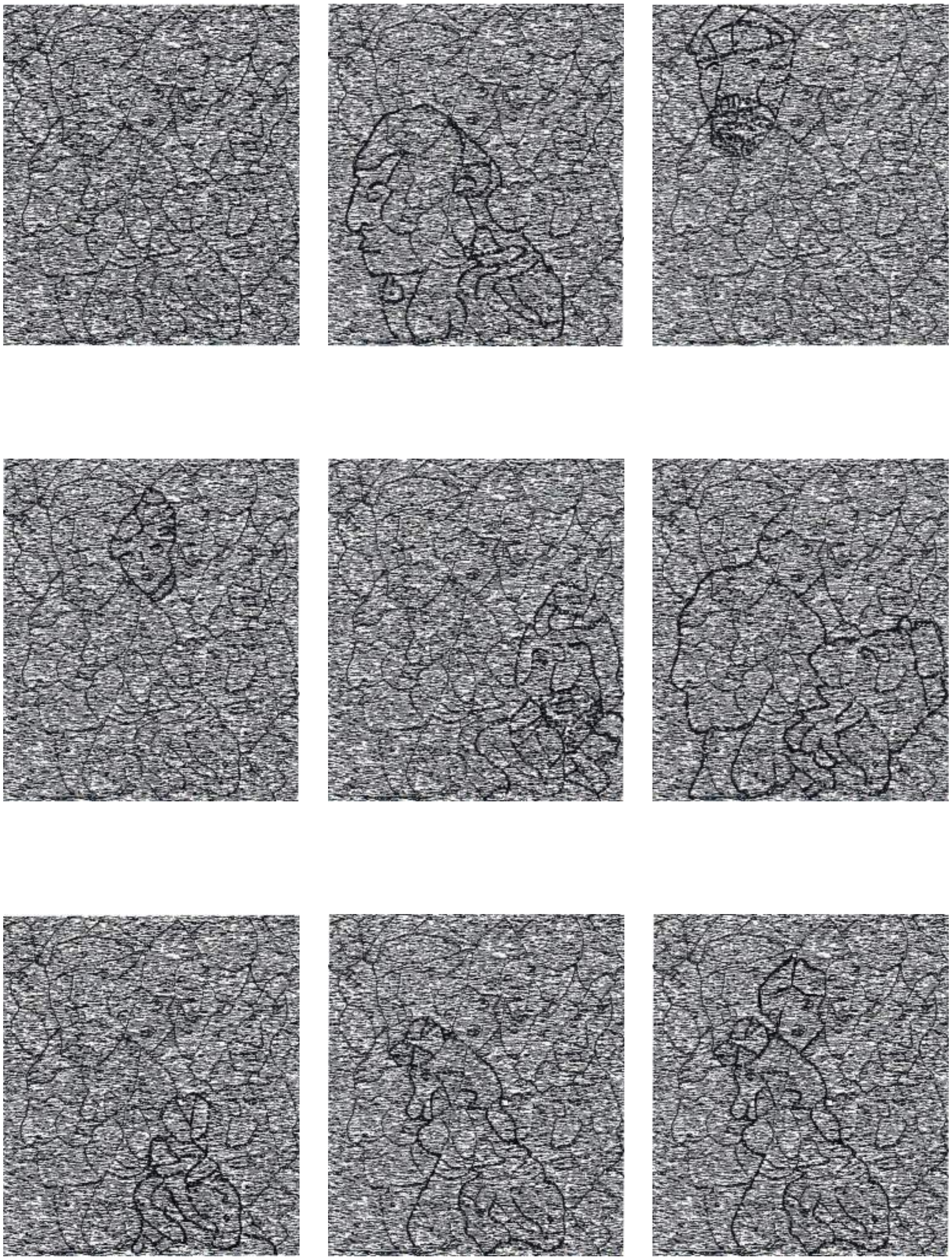

DRAFT





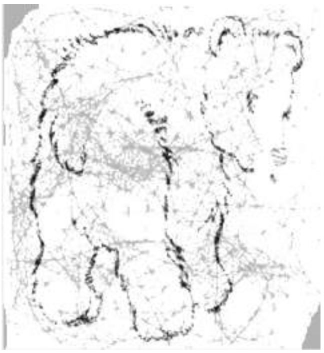


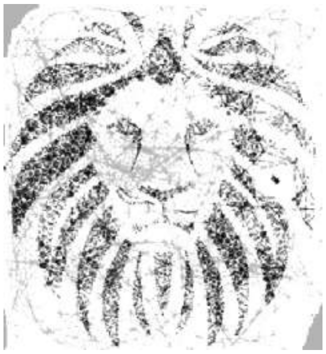


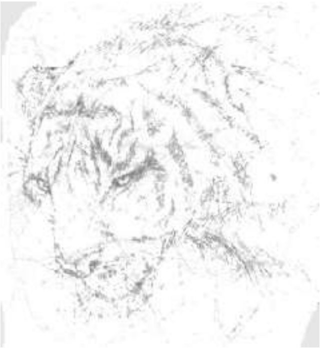
Appendix C

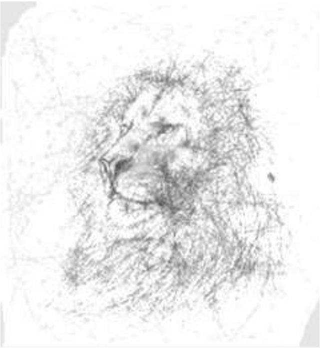
Examples of images obtained by using Random Masks

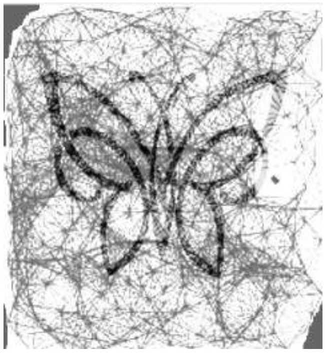
The following figures were obtained by using the Algorithms MGA 6.2.1 and ATTACH 7.2.1.

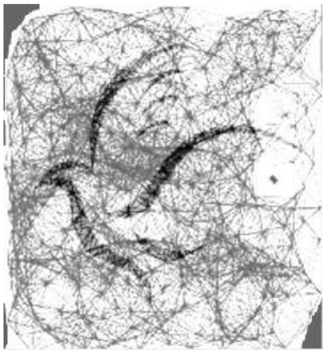




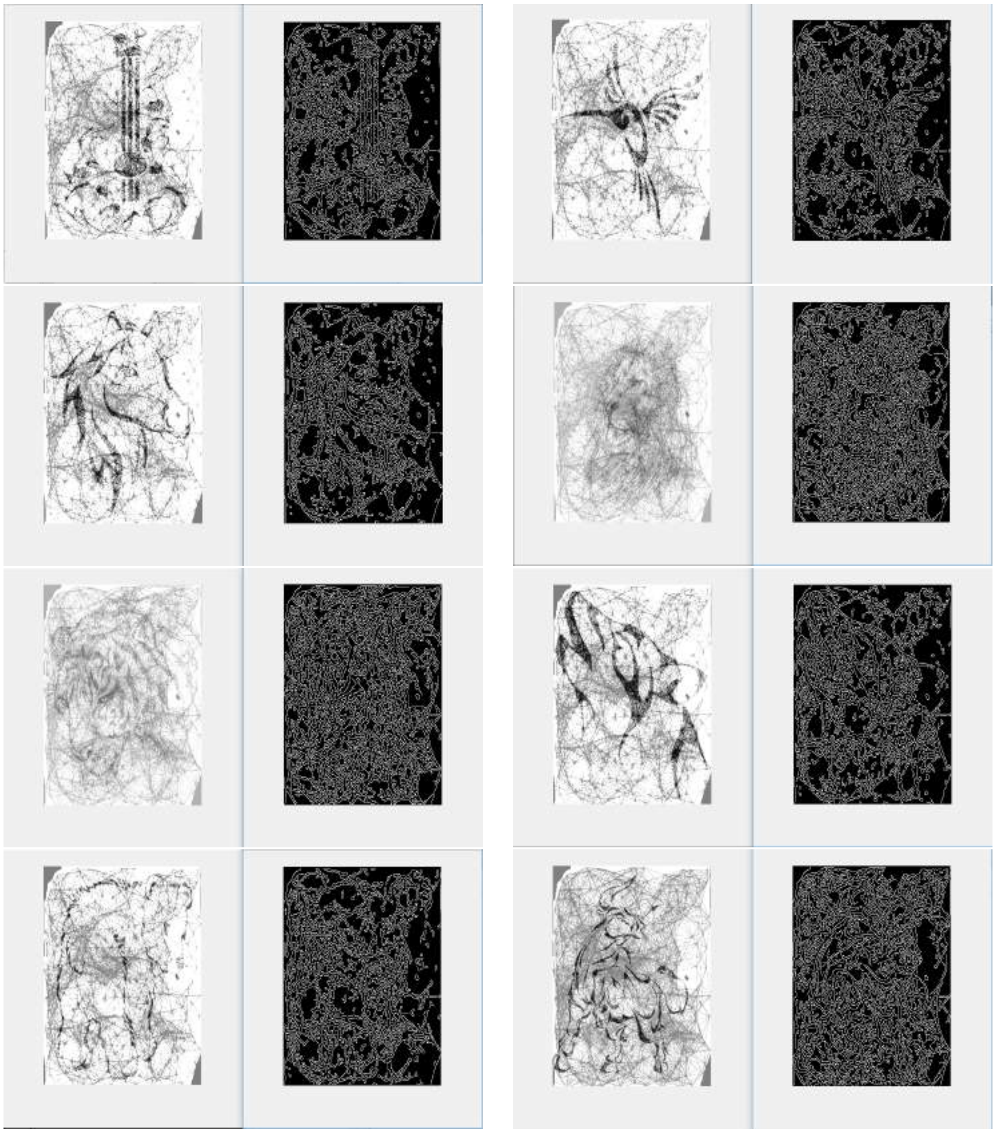








The following images show images obtained by using the Algorithms MGA 6.2.1 and ATTACH 7.2.1, and which have been tested by the Canny algorithm.



Bibliography

[1] L. Von Ahn, M. Blum, N.J. Hopper, and J. Langford, *CAPTCHA:Using hard AI problems for security*, In: Biham E. (eds) Advances in Criptology - EUROCRYPT 2003 ; Lecture Notes in Comput. Sci. **2656** (2003), 294–311. Springer, Berlin, Heidelberg.

[2] L. Von Ahn, M. Blum, and J. Langford, *Telling humans and computers apart automatically*, Commun. ACM **47** (2004), no. 2, 57-60.

[3] L. Von Ahn, B. Maurer, C. McMillen, D. Abraham, and M. Blum, *reCAPTCHA: Human-Based Character Recognition via Web Security Measures*, Science **321** (2008), 1465–1468.

[4] Altavista, *Altavista's "add-url" site, protected by the earliest known captcha.* http://altavista.com/sites/addurl/newurl, 1997.

[5] M.A.O. Angarita, E. Izquierdo, and A.M.Cañadas, *Human Interaction Proofs (HIPs) based on Emerging Images and Topological Data Analysis (TDA) Techniques*, 3rd Cyber Security in Networking Conference **to appear** (2019). Accepted.

[6] ______, *Human Interaction Proofs (HIPs) Based on Multistable Images and Brauer Configuration Algebras (BCA)*, 9th International Conference on Imaging for Crime Detection and Prevention (ICDP-19) **to appear** (2019). Accepted.

[7] M.A.O. Angarita, A.M.Cañadas, and E. Izquierdo, *Algebraic Tools for Multimedia Based Cryptography and Security Applications*,) **to appear** (2019). Submitted.

[8] M.A.O. Angarita and A.M.Cañadas, *Brauer Configuration Algebras for Multimedia Based Cryptography and Security Applications*,) **to appear** (2019). Submitted.

[9] G. Arce, Z. Wang, and G. Di Crescenzo, *Visual Cryptography from Halftone Error Diffusion: In: S. Cimato and C. Yang (eds)Visual Cryptography and Secret Image Sharing*, CRC Press. London, 2012.

[10] D.M. Arnold, *Abelian Groups and Representations of Finite Partially Ordered Sets*, CMS Books in Mathematics. vol. 2, Springer, 2000, 244 p.

[11] I. Assem, D. Simson, and A. Skowroński, *Elements of the Representation Theory of Associative Algebras*, Vol. 1, Cambridge University Press, 2006.

[12] H. S. Baird and K. Popat, *In Proceedings of the 5th International Workshop on Document Analysis Systems*, Springer-Verlag, 2002.

[13] H. S. Baird, M. A. Moll, and S. Y. Wang, *ScatterType: a legible but hard-to-segment CAPTCHA*, ICDAR'05, Proc. 8th Int. Conference on Document Analysis and Recognition, IEEE Computer Society (2005), doi: 10.1109/ICDAR.2005.205.

[14] S. Battiato, G. Gallo, and G. Puglisi, *Digital Imaging for Cultural Heritage Preservation: Analysis, Restoration*: *Digital Reproduction of Ancient Mosaics.*

[15] S. Battiato, G. Di Blasi, G. Farinella, and G. Gallo, *Digital Mosaic Frameworks-An Overwiew*, Computer Graphics forum. DOI:10.1111/j.1467-8659.2007.01021.x.

[16] W.A. Beyer, N. Metropolis, and J.R. Neergaard, *Statistical study of digits of some square roots of integers in various bases*, Math. Comput. **24** (1970), no. 110, 455-473.

[17] G. Di Blasi, G. Gallo, M. Petralia, and G. Gallo, *Puzzle Image Mosaic*, Proc. IASTED/VIIP2005 (2005).

[18] M. Blum, J. Langford, and N. Hopper, *The captcha project, "completely automatic public turing test to tell computers and humans apart"*, School of Computer Science. Carnegie-Mellon University, http://www. captcha. net, 2000.

[19] C. Blundo, P. D'Arco, A. De Santis, and D.R. Stinson, *Contrast optimal threshold visual cryptography schemes*, SIAM J. Discrete Math. **16** (2003), no. 2, 224-261.

[20] B. Bocian, *Fritz Perls in Berlin, 1893-1933: Expressionismus, Psychoanalyse, Judentum*, EHP Verlag Andreas Kohlhage, 2010.

[21] P. Borwein and L. Jörgenson, *Visible structures in Number Theory*, A.M.M **108** (2002), no. 5, 897-910.

[22] V. Buttigieg, *Variable-Length Error-Correcting Codes*, Doctoral Thesis, 1995.

[23] A.M. Cañadas and N.P.P. Vanegas, *Representations of Posets to Generate Emerging Images*, Far East J. Math. Sci. (FJMS) **Special** (2013), no. II, 139-152.

[24] A.M. Cañadas, I.D. Marín, and P.F.F. Espinosa, *Categorical properties of the algorithm of differentiation VIII, and on the algorithm of differentiation DIX for equipped posets*, JP J. Algebra Number Theory Appl. **29** (2013), no. 2, 133-173.

[25] A.M. Cañadas, H. Giraldo, and P.F.F. Espinosa, *Categorification of some integer sequences*, Far East J. Math. Sci. (FJMS) **92** (2014), no. 2, 125-139.

[26] A.M. Cañadas, M.A.O. Angarita, and W.G. Salas-Avila, *Matrix Problems to Generate Mosaic-Based CAPTCHAs*, 6th International Conference on Imaging for Crime Prevention and Detection (ICDP-15) (2015). London.

[27] A.M. Cañadas, R.J. Serna, and C.I. Espinosa, *On the reduction of some tiled orders*, JP J. Algebra Number Theory Appl. **36** (2015), no. 2, 157-176.

[28] A.M. Cañadas, V. Cifuentes, and A.F. Gonzalez, *On the number of two-point antichains in the powerset of an n-element set ordered by inclusion*, JP J. Algebra Number Theory Appl. **38** (2016), no. 3.

[29] A.M. Cañadas, R.J. Serna, and H. Giraldo, *Matrix problems induced by visual cryptography schemes*, Far East J. Math. Sci. (FJMS) **106** (2017), no. 2, 1223-1241.

[30] G. Carlsson and V. de Silva, *Topological estimation using witness complexes*, Symposium on PointBased Graphics (2004).

[31] S. Chakrabarti and M. Singbal, *Password-based authentication: Preventing dictionary attacks*, IEEE Computer **40** (2007), no. 6, 68-74.

[32] M. Chandrasekaran, R. Chinchani, and S. Upadhyaya, *Phoney: Mimicking user response to detect phishing attacks*, In Proceedings of the 2006 International Symposium on a World of Wireless, Mobile and Multimedia Networks, 668-672. IEEE, 2006.

[33] K. Chellapilla, K. Larson, P. Simard, and M. Czerwinski, *Building Segmentation based human-friendly human interaction proofs (HIPs)*, In H. Baird and D. Lopresti,edts, Human Interactive Proofs, Volume 3517 of Lecture Notes in Computer Sciencies, 173-185. Springer Berlin / Heidelberg, 2005.

[34] ———, *Designing human friendly human interaction proofs (HIPs)*, In Proceedings of the SIGCHI conference on Human factors in computing systems, 711-720. ACM, 2005.

[35] M. Chew and H. Baird, *BaffleText: a Human Interactive proof*, Proceedings of the SPIE/ISET Document Recognition and Retrieval Conf (2003), 22-23. Santa Clara.

[36] D. Cohen-Steiner, H. Edelsbrunner, and J. Harer, *Stability of Persistence Diagrams*, Discrete Comput. Geom. **37** (2007), no. 1, 103-120. https://doi.org/10.1007/s00454-006-1276-5.

[37] B.A. Davey and H.A. Priestley, *Introduction to Lattices and Order*, Second edition, Cambridge University Press. Cambridge, 2002.

[38] A. Desolneaux, L. Moisan, and J. M. Morel, *Gestalt Theory to Image Analysis: A Probabilistic Approach*, 2006.

[39] J. Dobashi, T. Haga, H. Johan, and T. Nishita, *A Method for creating mosaic images using Voronoi Diagrams*, Computer Graphics Forum (EG'02), 341-348. 2002.

[40] H. Edelsbrunner, J. Harer, and A. Zomorodian, *Topological Persistence and Simplification*, Discrete Comput. Geom. **28** (2002), 511–533.

[41] H. Edelsbrunner and J. Harer, *Persistent Homology - a survey*, In Twenty Years After. Eds. J.E. Goodman, J. Pach and R. Pollack. AMS, 2007.

[42] ———, *Computational Topology, an Introduction*, Amer. Math. Soc. Providence, Rhode Island, 2009.

[43] H. Edelsbrunner, *Persistent Homology in Image Processing*, In: W.G.Kropatsch, N.M. Artner, Y. Haxhimusa, X. Jiang (eds). Graph-Based Representations in Pattern Recognition GbRPR. Lecture Notes in Comput. Sci., vol 7877. Springer, Berlin, Heidelberg, 2013.

[44] W. Ellis, *A source book of Gestalt psychology*, Vol. 2, Routledge. Taylor y Francis Group, 1938. Great Britain.

[45] E. Fernández and M.I. Platzeck, *A note on the spectral properties of cluster algebras* (2010). Preprint available in: https://arxiv.org/abs/1011.5520v1.

[46] I. Fischer and T. Herfet, *Visual captchas for document authentication*, In IEEE 8th Workshop on Multimedia Signal Processing, 471–474. IEEE, 2006.

[47] L. Fritzsche, H. Hellwigt, S. Hiller, and O. Deussen, *Interactive design of authentic looking mosaics using Voronoi structures*, In 2. International Symposium on Voronoi Diagrams in Science and Engineering (VD). 2005.

[48] P. Gabriel and A.V. Roiter, *Representations of Finite Dimensional Algebras*: *Algebra VIII, Encyclopedia of Math. Sc.*, Vol. 73, Springer-Verlag, 1992. 177p.

[49] H. Gao, D. Yao, H. Liu, and L. Wang, *A novel Image based CAPTCHA using jigsaw puzzle*, In Computational Science and Engineering (CSE),2010 IEEE 13th International Conf., 351-356. dec. 2010.

[50] S. Gao, M. Mohamed, N. Saxena, and C. Zhang, *Emerging-Image Motion CAPTCHAs: Vulnerabilities of Existing Designs, and Countermeasures*, 2015. http://dx.doi.org/10.1145/2818000.2818006.

[51] ______, *Emerging Image Game CAPTCHAs for Resisting Automated and Human-Solver relay Attacks*, ACSAC'15 (2015). http://dx.doi.org/10.1109/TDSC.2017.2719031.

[52] R. Ghrist, *Barcodes: The persistent topology of data*, Bull. Amer. Math. Soc. **45** (2008), 45-61. http://www.ams.org/bull/2008-45-01/S0273-0979-07-01191-3. 4.

[53] P. Golle and N. Ducheneaut, *Preventing bots from playing online games*, Computers in Entertainment (CIE) **3** (2005), no. 3, 3–3.

[54] E.L. Green and S. Schroll, *Brauer configuration algebras: A generalization of Brauer graph algebras*, Bull. Sci. Math. **141** (2017), 539–572.

[55] M. Hazewinkel, N. Gubareni, and V.V. Kirichenko, *Algebras, Rings and Modules*, First Edition, Vol. 2, Springer, 2007.

[56] G. Humphrey, *The Psychology of the Gestalt*, J. Educ. Psychol. **15** (1924), no. 7, 401.

[57] G. Kanizsa, *Grammatica del vedere / La Grammaire du voir*, Il Mulino Bologna / Éditions Diderot, arts et sciences, 1980/1997.

[58] E.J. Kartaltepe and S. Xi, *Towards blocking outgoing malicious impostor emails*, In Proceedings of the 2006 International Symposium on a World of Wireless, Mobile and Multimedia Networks, 5-pp. IEEE, 2006.

[59] M.M. Kleiner, *Partially ordered sets of finite type*, Zap. Nauchn. Semin. LOMI **28** (1972), 32–41 (in Russian); English transl., J. Sov. Math **3** (1975), no. 5, 607–615.

[60] K. Koffka, *Principles of Gestalt Psychology*, Routledge, 2005.

[61] Y.M. Kuo, H.K. Chu, M.T. chi, R.R. Lee, and T.Y. Lee, *Generating Ambiguous Figure-Ground Images*, IEEE Trans. Vis. Comput. Graph. **23** (2017), no. 5, 1534-1545.

[62] S. Lehar, *Gestalt isomorphism and the primacy of subjective conscious experience: A Gestalt Bubble model*, Behav. Brain Sci **26** (2003), no. 4, 375-408.

[63] S. Ming-Shing, H. Wen-Liang, and C. Kuo-Young, *Digital imaging for cultural Heritage preservation: Analysis, Restoration*, IEEE Transactions on Images Processing focuses and signal-processing aspects of image processing, imaging systems, and images scanning, display and printing) **13**, 952–959. IEEE, 2007.

[64] N. J. Mitra, H. K. Chu, T. Y. Lee, L. Wolf, H. Yeshurun, and D. Cohen-Or, *Emerging Images*, ACM Trans. Graph. **28** (December, 2009), no. 5, 8 pp.

[65] P. Moore and Ch. Fitz, *Using Gestalt Theory to Teach Document Design and Graphics*, Tech. Commun. Q. **2** (Fall 1993), no. 4, 389-410.

[66] G. Mori and J. Malik, *Recognizing objects in adversarial clutter: Breaking a visual CAPTCHA*, Proc. Conf. Computer vision and pattern recognition (2003). Madison.

[67] M. Naor and A. Shamir, *Visual Cryptography*, Advances in Cryptography: EUROCRYPT'94; Lecture Notes in Comput. Sci. **950** (1994). Springer, Berlin, Heidelberg.

[68] M. Naor, *Verification of a human in the loop or identification via the Turing test* (September 13 1996). Unpublished notes; http://www.wisdom.weizmann.ac.il./naor/PAPERS/human.pdf.

[69] M. Nayeem, M. Akand, N. Sakib, and W. Kabir, *Design of a Human Interaction Proof (HIP) using human cognition in contextual natural conversation*, 2014 IEEE 13th International Conference on Cognitive Informatics and Cognitive Computing (2014), 146–154.

[70] L.A. Nazarova and A.V. Roiter, *Representations of partially ordered sets*, Zap. Nauchn. Semin. LOMI **28** (1972), 5–31 (in Russian); English transl., J. Sov. Math. **3** (1975), 585–606.

[71] U. Onwudebelu, U. Ugwuoke, and I. Nkechi, *A Review and Evaluation of Human Interactive Proof (HIP) Technique for Combating Malicious Automated Scripts*, Computer Science and Information Technology **1** (2013), no. 3, 202-207.

[72] A. Pinar Saygin, I. Cicekli, and V. Akman, *Turing test: 50 years later*, Minds Mach. **10** (2000), no. 4, 463–518.

[73] G. Reynaga, *The usability of CAPTCHAS on mobile devices*, Carleton University, 2015. Ottawa, Ontario.

[74] C. Rodriguez and A.G. Zavadskij, *On corepresentations of equipped posets and their differentiation*, Revista Colombiana de Matemáticas **39** (2006).

[75] C. Romero-Macias and E. Izquierdo, *Image CAPTCHA based on distorted faces*, 4th international conference on imaging for crime prevention and detection; IEEE digital library (2011), doi: 10.1049/ic.2011.0106. London.

[76] ______, *A survey of captchas: Are computers getting the better of us?*, ACM Surveys (2011).

[77] C. Romero-Macias, *Image Understanding for Automatic Human and Machine Separation*, Queen Mary, University of London. School of Electronic Engineering and Computer Science (2013). PhD Thesis. London.

[78] Y. Rui and Z. Liu, *ARTiFACIAL: Automated Reverse Turing test using FACIAL features*, Proceedings of ACM Multimedia 2003, (2003).

[79] S. Saklikar and S. Saha, *Public key-embedded graphic captchas*, In 5th IEEE Consumer Communications and Networking Conference (CCNC), 262–266. IEEE, 2008.

[80] B.S.W. Schröder, *Ordered Sets. An Introduction*, Birkhäuser Boston, Inc.,. Boston, MA,2003.

[81] S. Shirali-Shahreza and A. Movaghar, *A new anti-spam protocol using captcha*, In IEEE International Conference on Networking, Sensing and Control (2007), 234–238. IEEE.

[82] S. Shirali-Shahreza, M. Shirali-Shahreza, and A. Movaghar, *Restricted access to exam grades on the web by HIP*, In 6th IEEE/ACIS International Conference on Computer and Information Science (ICIS), 967–971. IEEE, 2007.

[83] S. Shirali-Shahreza and M. Shirali-Shahreza, *A Survey of Human Interactive Proofs Systems*, Int. J. Innov. Comput. **6** (March 2010), no. 3A.

[84] D. Simson, *Linear Representations of Partially Ordered Sets and Vector Space Categories*, Gordon and Breach, 1992. London.

[85] M. Sonka, V. Hlavak, and R. Boyle, *Image Processing, Analysis and Machine Learning*, 2nd edn, PSW Publishing. Pacific Grove, 1999.

[86] P. Sterzer, A. Kleinschmidt, and G. Rees, *The neural bases of multistable perception*, Trends Cogn. Sci. **13** (2009), no. 7, 310–318. doi: http://dx.doi.org/10.1016/j.tics.2009.04.006.

[87] D.R. Stinson, *Cryptography: Theory and Practice*, First Edition, CRC Press, 1995.

[88] http://www.usm.edu/media/english/fairytales/cinderella/imagesonly.html.

[89] techopediA, *Internet Bot* (2018). https://www.techopedia.com/definition/24063/internet-bot.

[90] Q. Tong, S. H. Zhang, R. R. Martin, and P. L. Rosin, *Nested Images*, Preprint.

[91] Q. Tong, S. Zhang, H. Martin, R. Ralph, and P. Rosin, *Nested Images*, Presented at: Asian Conference on Design and Digital Engineering 2011 (ACDDE 2011) (August, 2011), 445-450.

[92] J. Tsotsos, *On the Relative Complexity of Active Vs. Passive Visual Search*, Int. J. Comput. Vis. **7** (1992), no. 2, 127-141.

[93] A. M. Turing, *Computing Machinery and Intelligence*, Mind **49** (1950), 433-460.

[94] S. Weinberger, *What is $\cdots$ Persistent Homology ?*, Notices of AMS **58** (2011), no. 1, 36-39.

[95] S. Woo, *Design and evaluation of 3D CAPTCHAS*, Computers & Security **82** (2019), 49-67. https://doi.org/10.1016/j.cose.2018.12.006.

[96] Y. Xu, G. Reynaga, S. Chiasson, J. Frahm, F. Monrose, and P.Van Oorschot, *Security analysis and related usability of motion-based captchas: Decoding codewords in motion*, IEEE Transactions On Dependable And Secure Computing **11**, no. 5. 2013.

[97] Y. Xu, G. Reynaga, S. Chiasson, J-M. Frahm, F. Monrose, and P. Van Oorschot, *Security and usability challenges of moving-object CAPTCHAs: decoding codewords in motion*, Proceeding Security'12 Proceedings of the 21st USENIX conference on Security symposium. IEEE TDSC **11** (2014), no. 5, 480–493. doi: 10.1109/TDSC.2013.52.

[98] R.V. Yampolskiy, *Detecting and controlling cheating in online poker*, In 5th IEEE Consumer Communications and Networking Conference (CCNC). IEEE,2008.

[99] A.G. Zavadskij, *On Two-Point Differentiation and its Generalization*, Contemp. Math. **376** (2005).

[100] A.G. Zavadskij and V.V. Kirichenko, *Semimaximal Rings of finite type*, Mat. Sb. (N.S.) **103** (1977), 323-345. (in Russian).

[101] ———, *Torsion-free modules over primary rings*, J. Sov. Math. **11** (1979), no. 4, 598-612.

[102] X. Zhu, *Persistent Homology: An Introduction and a New Text Representation for Natural Language Processing*, AAAI Press. Beijing, China, August 3-9, 2013.

[103] B. Zhu, J. Yan, Q. Li, C. Yang, J. Liu, N. Xu, M. Yi, and K. Cai, *Attacks and Design of Image Recognition CAPTCHAs*, CCS 2010 ACM (2010).

[104] A. J. Zomorodian, *Topology for Computing*, Cambridge Monographs on Applied and Computational Mathematics (2005). Cambridge University Press, http://www.cs.dartmouth.edu/ afra/book.html. 9,25.

[105] A. Zomorodian and G. Carlsson, *Computing Persistent Homology*, Discrete Comput. Geom. **33** (2005), 249-274.