The Kaleidoscopic Game of Life

Document Version:
Publisher's PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:

• A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

Download date: 12. Jan. 2025

The Kaleidoscopic Game of Life

Aida Abiad

Department of Mathematics and Computer Science
Eindhoven University of Technology
Eindhoven, The Netherlands

Department of Mathematics: Analysis, Logic and Discrete Mathematics
Ghent University
Ghent, Belgium

Department of Mathematics and Data Science
Vrije Universiteit Brussel
Brussels, Belgium
a.abiad.monge@tue.nl

Merit Geldmacher
Alexander Grigoriev

Department of Data Analytics and Digitalization
Maastricht University
Maastricht, The Netherlands
merit.geldmacher@gmail.com
a.grigoriev@maastrichtuniversity.nl

Dedicated to the memory of John Conway

Several image rotation operators are introduced as novel rotation variants of John Conway's Game of Life. The main idea of the Game of Life is to simulate real-life processes such as birth, survival and death. The discrete model operates on a two-dimensional array of black and white square cells. In this paper, we study the development of patterns in image rotation using different rotation methods. Moreover, the quality of the rotation and the applicability of image rotation to the Game of Life are investigated.

1. Introduction

The Game of Life is a widely studied and well-known cellular automaton that was developed by John Horton Conway in the late 1960s and popularized by Gardner in [1]. Even today, about 50 years after its invention, it still enjoys a great deal of attention from researchers. The Game of Life is defined as a discrete model consisting of a cell grid where each cell has a finite number of dimensions and a finite

number of states. Furthermore, the state of each cell at time t is determined by a function of a certain set of cells. This function depends on the states of a finite selection of cells (referred to as the neighborhood) at time $t-1$, where time is modeled discretely.

In the literature [2], a cellular automaton is classified as lifelike if five characteristic criteria are met. The first criterion is that the cells must form a two-dimensional square lattice and a cell's neighborhood consists of the eight cells that are orthogonally and diagonally adjacent to it. Additionally, each cell of the automaton may be in one of two possible states, live or dead. In every time step of a sequence of time steps, the status is updated simultaneously for all cells. The update decision of a given cell at time t is based on a function that inputs the state of that cell at time $t-1$ and the number of live neighbors at time $t-1$. Consequently, every cell has the same updating rule and after every simultaneous update of all cells, a new generation is created. According to those lifelike criteria, Conway's Game of Life is such a cellular automaton. See for instance [2].

The Game of Life is a zero-player never-terminating game that is solely determined by its initial state. Its charm emerges from observing the cells on the grid proliferate. This entails but is not limited to the convergence to an equilibrium such as the steady state or the appearance of cycles. Since the game's invention, many variations and applications have been developed. An example of a non-lifelike game is Bays' game, which adds a third dimension to the grid and allows a cell to interact with more than eight neighbors [3, 4]. Bays also conducted research on playing the Game of Life on triangular, pentagonal and hexagonal tessellations where, for instance, each triangle on the triangular tessellation has 12 neighboring triangles; see [5–7].

Furthermore, there exist versions with different rules on the number of live or dead states that decide whether a cell is born, survives or dies. Eppstein [2] investigated and compared the behavior of such rules. Levene and Roussos [8] enhanced the Game of Life to a two-player game with competitive rule elements for birth and survival. The Game of Life is extended with competitive elements in Lauer et al.'s [9] work on research optimization and self-interest, where the model is amplified by the well-known prisoner's dilemma.

The Game of Life is not only altered or extended by various models, but additionally, there are other examples where the original Game of Life is used as a tool within other algorithms. For example, Khankasikam's research [10] deals with Lanna character recognition, and the Game of Life framework gets integrated into the genetic algorithm for character recognition. Recently, Aguilera-Venegas et al. [11] introduced probabilistic cellular automata that include nondeterministic rules for transitions between successive generations of the automaton, together with probabilistic decisions about life and death of the

cells in the next generation of the automaton. Several other variations of the Game of Life have also been recently considered; see for instance [12–14].

Those are just a few examples of variations, extensions and applications of the Game of Life. This paper aims at introducing a new variant of the game that interlinks image rotation methods with the Game of Life. So far, those two concepts have not been investigated together. The rules of the Game of Life are substituted with rotation operators. The main goal of this paper is to compare and evaluate different rotation and visualization methods, especially with respect to their long-run behavior, their stable states and their ability to develop recognizable patterns such as twirls, balls and rings. Additionally, the research should confirm or reject whether image rotation is a valid rule set for the Game of Life.

This paper is structured as follows. Section 2 introduces and defines the Game of Life as well as the concept of image rotation. Furthermore, the assumptions and definitions regarding image rotation are specified. Section 3 introduces the methodology, which consists of rotation procedures and visualization procedures. Sections 4 and 5 describe, interpret and discuss the findings of the simulations. The results for several rotation angles are presented and potential drawbacks of the method are identified. Finally, Section 6 presents a summary as well as a discussion of all empirical and theoretical findings and proposes further research. The implementation of the models has been made in R.

2. Preliminaries

2.1 The Game of Life

This section introduces the Game of Life and its rules based on Gardner's work [1]. The Game of Life belongs to the class of simulation games since it resembles real-life processes. It is played on an infinite orthogonal two-dimensional grid consisting of square cells. Each cell is in exactly one out of two available states; live or dead, which is visualized by the colors black and white, respectively.

The main idea is to start with some initial pattern and to observe how it changes when the rules are applied iteratively. John Conway, the inventor of the Game of Life, chose the rules to meet three requirements [1]:

1. There should not exist an initial pattern for which it can easily be proven that the population can grow infinitely.

2. There should exist, however, initial patterns that seem to grow infinitely.

3. There should exist simple initial patterns that first develop and change for a certain amount of time before they fade away entirely (too sparse or too overcrowded), converge to a stable configuration or settle into a finite-length cycle of at least two periods.

The result of those requirements is that the development of the population becomes unpredictable. With the initial pattern being the seed of the system, each further generation is generated by a cell's interaction with its eight adjacent neighbor cells. The simultaneous application of the rules determines the transition from one generation to the next. The specific rules as invented by Conway are:

1. Any live cell with fewer than two live neighbors dies during the transition to the next generation.

2. Any live cell with two or three live neighbors stays live during the transition to the next generation.

3. Any live cell with more than three live neighbors dies during the transition to the next generation.

4. Any dead cell with exactly three live neighbors becomes live during the transition to the next generation.

Those rules can be related to death by underpopulation, sustainable life, death by overpopulation and birth, respectively, and thus motivate the Game of Life.

Common patterns that are observed in the Game of Life are still lives, oscillators and spaceships. Still lives do not change during the transition from one generation to the next. A still life is sometimes defined to have period one; however, during this paper, the expressions of still life or convergence are used. An oscillator is a pattern that is a predecessor of itself. It is a pattern that repeats itself after a fixed and finite (at least two) number of generations. This fixed number of time steps until reoccurrence is defined as its period. Spaceships return to the same configuration after a finite number of generations as well, but located elsewhere on the grid. In other words, a spaceship can be described as an oscillator that travels across the grid.

2.2 Image Rotation

The geometric concept of rotation describes a motion of a given space that maintains at least one point. The image rotation considered in this article is the rotation of an image around its center pixel. Furthermore, images are assumed to be displayed on a finite orthogonal grid of pixel cells. When rotating such an image by a given angle, it cannot be avoided that information is lost due to the combination of the discrete grid and the continuous rotation operation. Let an image on a fixed two-dimensional grid of same-size square cells be rotated around the center by 10 degrees. The issue is that the square pixels of the image do not perfectly get mapped on the fixed cells of the grid.

Consequently, the rotated pixels of the image and the square cells of the grid overlap and the image cannot be displayed without damage. This holds true for all rotation angles that are not a multiple of 90 degrees. This issue is in practice often addressed by smoothing with an area-weighted average of colors. Such methods work sufficiently well for high-resolution images, however, they still create damage to the image.

One goal of this paper is to compare the long-run behavior of different rotation operators. The long-run behavior entails the study of equilibria and whether the pattern is able to survive, or may even return to its original.

2.2.1 The Grid

The image rotation is performed on a finite two-dimensional orthogonal grid of square cells. Each cell has a width and a height of one and the center of a cell corresponds to a point on the grid with integer coordinates. This implies that the corners of a pixel have half-integer coordinates. Figure 1 states an example. The coordinates of that pixel are $(0, 0)$, which corresponds to the center of the pixel. The corners all have half-integer coordinates, either half a unit above or half a unit below the center coordinates. The enumeration of pixels in this paper is such that the neighbor pixel on the top has coordinates $(0,1)$. The enumeration of the neighbors continuing clockwise is $(1, 1)$, $(1, 0)$, $(1, -1)$, $(0, -1)$, $(-1, -1)$, $(-1, 0)$ and $(-1, 1)$.
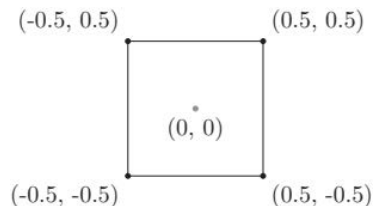

Figure 1. Example of the center pixel on a grid.

2.2.2 The Images

Within this research, only square images are considered; that is, the number of rows of horizontally aligned pixels equals the number of columns of vertically aligned pixels. Furthermore, the total number of rows and the total number of columns of the image are assumed to be odd. This ensures the existence of one center pixel. However, if an image does not have the mentioned properties, it could be adapted by adding a few rows and/or columns of white pixels such that the number of rows and columns is odd and a square grid is ensured. This is not an issue in this paper. Two types of images are considered,

one-color (black pixels and empty pixels) and two-color images (black and white pixels). Within this paper, the empty pixels are visualized as white cells. In the one-color case, only black pixels undergo the rotation procedure, whereas in the two-color case, both black and white pixels are rotated.

The implementation of such images and the rotation procedures have been done using the programming language R.

2.2.3 The Rotation Matrix

The rotation method makes use of the commonly used rotation matrix. Consider the matrix that rotates a given vector x in $\mathbb{R}^2$ clockwise around the origin by α degrees in the Cartesian coordinate system. Then,

$$x' = \begin{bmatrix} x'_1 \\ x'_2 \end{bmatrix} = \begin{bmatrix} \cos(\alpha) & \sin(\alpha) \\ -\sin(\alpha) & \cos(\alpha) \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{1}$$

where the matrix

$$\begin{bmatrix} \cos(\alpha) & \sin(\alpha) \\ -\sin(\alpha) & \cos(\alpha) \end{bmatrix} \tag{2}$$

is known as the *rotation matrix*. Note that $x = [x_1, x_2]$ is the original point and $x' = [x'_1, x'_2]$ is the rotated point. Neither of the two points x and x' must generally be an integer. For example, rotating point (5, 6) clockwise by 30 degrees around the center gives coordinates $\cos(30) \cdot 5 + \sin(30) \cdot 6 \approx 7.33$ and $-\sin(30) \cdot 5 + \cos(30) \cdot 6 \approx 2.70$.

If an image is rotated around a fixed angle more than once, a sequence of rotations is created. A sequence of rotations is defined as *cyclic* if successive rotations around the same angle result in an infinite fixed-length recurring series of images. For example, if in a series of images every second image is the same, such as $(a, b, a, b, a, ...)$, the series is classified as a cycle of length two. Note that the period of such a cycle is required to be at least two. If the image does not change anymore when the rotation operator is applied again, the series of images has converged.

2.2.4 Assumptions

For the remainder of this paper, square images with an odd number of rows and columns are assumed, implying the existence of one center pixel with coordinates (0, 0). The remaining pixels have locations as in the Cartesian coordinate system. The integer coordinates refer to the center of a square pixel. Rotation of a pixel is defined as the rotation of a vector between the origin and the center of that pixel, which

gets mapped on another vector between the origin and the center of the rotated pixel. That pixel provides the coordinates of the rotated pixel. Rotation is always performed around the center pixel, which can be compared to the origin of the Cartesian coordinate system.

For simplicity, only angles of α degrees that are divisors of 360 degrees are considered, so $360 \bmod \alpha$ equals zero. The 24 divisors of 360 are 1, 2, 3, 4, 5, 6, 8, 9, 10, 12, 15, 18, 20, 24, 30, 36, 40, 45, 60, 72, 90, 120, 180 and 360. Note that, however, the methods themselves are in no way restricted to those rotation angles. Furthermore, given an angle of α degrees, a full rotation is undertaken if an image is rotated $360 / \alpha$ times by α degrees, and thus, completes the full circle.

The analysis of the rotation results includes connectedness and disconnectedness of different black components. Two black components are considered *connected* if they share at least one pixel border. Therefore, the two black pixels in Figure 2 are disconnected from each other since they do not share a pixel border but only touch each other diagonally.
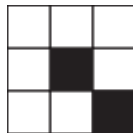

Figure 2. Example of two disconnected black pixels.

2.3 Parallels between the Game of Life and Image Rotation

There are several similarities between the Game of Life and image rotation. To begin with, the playing field of both concepts is a two-dimensional orthogonal grid of square cells, where each cell is in one of two possible states: 0 or 1, live or dead, black or white. When one-color images are considered, the terms black and empty pixels are used and when two-color images are considered, the terms black and white are used. Some patterns created by the Game of Life have the ability to move along the grid, whereas the image rotation is performed around a center pixel. Thus, the images are bound to the area relatively close to the center. In other words, when the images are rotated around the center successively, even though they could potentially explode outward in all directions, they cannot move over, for instance, to the right of the grid. Therefore, a finite grid is suitable for image rotation, whereas the Game of Life may require an infinite-size grid.

The operators are, by definition, another major difference between the Game of Life and image rotation. The Game of Life considers a very local operator that lets cells turn black, turn white or stay their color, depending on the states of the eight neighboring cells. The

rotation operator, on the other hand, combines global and local aspects, meaning that it shifts pixels around. This lets the local neighborhood of a cell after the rotation be very similar to how it is before the rotation; however, it is found in a different region of the grid. The location of the cells depends on the rotation angle as well as the previous location. Thus, a subset of the pixels might have a very similar local neighborhood but is found in a completely different region, considering the global scale.

An additional similarity between the two concepts is that cells can change their color. In the Game of Life, cells change their color (pixels die or get born) depending on their neighbor cells. Thus, for instance, when a specific cell changes its color from black to white, it dies. The change of states is with respect to a specific cell on the grid. During the image rotation, however, a black pixel dies when it is not assigned to any pixel in the visualized image. In other words, it dies when no pixel in the visualized image becomes black because of that rotated black pixel. This can happen, for instance, when two black pixels get rotated onto the same cell on the pixel grid. On the other hand, pixels get born when one rotated black pixel causes two or more pixels in the visualized image to be black. This occurs, for example, when a black pixel gets mapped exactly on the intersection between two or more pixels. Furthermore, both the Game of Life and the image rotation are zero-player games, meaning that the development of the game is solely determined by its input or initial state. The image rotation further requires a rotation angle, and one of the visualization methods demands a parameter initialization before the rotation. However, once the angle and the parameter are set, the subsequent development is exclusively determined by the rotation process and is not influenced externally. The specific methods are explained in Section 3. Finally, both concepts incorporate the occurrence of patterns and their developments. Both the progress of certain patterns and the study of cycles and long-term behavior are addressed in this paper.

3. Methodology

We consider two alternating procedures. The first procedure is the rotation of a given image, and the second procedure is the visualization of the rotated image. In the experiments, the number of times the procedures are repeated depends on the rotation angle.

This section defines and describes the underlying rotation procedures and the different visualization methods that have been used in this article. In addition, it specifies criteria to compare and to evaluate the different methods.

▌ 3.1 The Rotation Procedures

Recall that depending on the rotation angle, the pixels' coordinates after a rotation step may not be integers. However, after the visualization process, all pixels must be either colored or empty. Thus, after visualization, all pixels have integer coordinates since each cell in the grid has identifying integer coordinates. For example, a black pixel may have coordinates $(5, 6)$ before rotation. Its coordinates after rotation are approximately $(5.97, 5.04)$ if the image is rotated around 10 degrees. During the visualization process, the pixel on the grid with coordinates $(6, 5)$ is likely to be assigned the color black since $(6, 5)$ is the closest integer coordinate point to $(5.97, 5.04)$. Clearly, the specific color assignment may depend on the visualization method.

3.1.1 Zero-Lag and One-Lag Rotation

Within the rotation, the two different procedures are one-lag rotation and zero-lag rotation. Both methods utilize the rotation matrix as their main instrument. However, the input for the rotation procedures, as well as the memory capability from one rotation to the next one, differs.

The input for the zero-lag rotation is pixels with integer coordinates and an associated color. Thus, for two successive rotations, the second rotation procedure takes the visualized version of the first rotation procedure as its input. This means that the zero-lag rotation method takes an image as the input and gives out rotated coordinates that may be noninteger. Then, it visualizes the image from those rotated coordinates and takes the resulting image as the input for the next rotation procedure.

Taking up the previous example again, it would look as follows. The black pixel with coordinates $(5, 6)$ has rotated coordinates $(5.97, 5.04)$. Assuming the pixel with coordinates $(6, 5)$ gets assigned black during the visualization process, the following zero-lag rotation procedure would take pixel $(6, 5)$ and the associated color black as an input.

The one-lag rotation, on the other hand, has a one-step memory capability with respect to the rotated pixel coordinates and colors. It can memorize the exact coordinates of the previous rotation operation and use them for the next rotation iteration. Thus, the visualization of the images serves as a progress status of the rotation. However, the visualized images are not used for the next rotation step.

Coming back to the example, the first rotation step would take $(5, 6)$ as input coordinates and get $(5.97, 5.04)$ as output coordinates. During the visualization procedure, pixel $(6, 5)$ may be colored black. For the next rotation, however, the input would be that the pixel at

location (5.97, 5.04) is black. The output of this rotation procedure would be the input for the third rotation, and so on. Thus, after visualizing the rotation results, the pixels in the intermediate images all have integer coordinates. However, the input for the next one-lag rotation iteration is the previous step's exact coordinates. This minimizes damage to ensure good rotation results.

Even though both the one-lag rotation and the zero-lag rotation are alternating processes of performing a rotation step and visualizing the result, the essential difference lies in the inputs for the rotation procedure. They take the previous step's rotation operation result and the previous step's visualized result, respectively. Part of this paper's research is to analyze and to evaluate the trade-off between the accuracy and memory requirement of the two rotation approaches.

▌ 3.2 The Visualization Procedures

Different visualization methods for two types of images have been developed and tested. The types of images distinguish themselves by the number of colors they contain. The first type of images is one-color images; given a pixel grid, the image is characterized by black pixels and empty pixels. Even though the empty pixels are visualized as white pixels in the plots within this paper, they are treated as empty pixels during the experiments. This favors the running time of the rotation and the visualization procedure since empty pixels neither get rotated nor visualized. On the other hand, two-color images do not contain any empty cells, but each pixel is either black or white.

In the experiments, black-only images are rotated using the black-only rotation method, which is a zero-lag rotation procedure. For simplicity, the zero-lag extension is dropped and the method is usually called black-only rotation rather than zero-lag black-only rotation. The images that are rotated by this method get visualized by two different methods: the *Min_dist* method and the *Variation* method, which get introduced in the next subsections. Furthermore, the black and white images get rotated by the zero-lag as well as the one-lag black and white rotation method. Thus, they are dealt with using two different rotation procedures. The visualization approach for black and white images, however, is the same for both rotation methods and is described in Section 3.2.3.

3.2.1 *Min_dist* Visualization Method for Zero-Lag Black-Only Rotation

This subsection deals with the visualization process of black-only images that have already undergone the zero-lag rotation procedure. Two approaches are defined, the *Min_dist* method and the *Variation* method. The input for both of the methods is a matrix with the coordinates of each black pixel of the rotated image. This matrix can

contain empty entries that correspond to the coordinates of the empty pixels. The methods' task is to visualize the image based on that matrix.

Min_dist. Algorithm 1 presents the first visualization procedure for black-only images. The main idea is that for each pixel of an empty pixel grid it should be decided whether it takes on the color black or stays empty. Those pixels on the empty grid are referred to as pixels on the grid. A pixel on the grid is colored black if any black pixel from the rotated image is close enough. Otherwise, the pixel on the grid stays empty. Close enough is defined by being in the range of *Min_dist* units.

Input: Coordinates of each black pixel of the rotated image
Output: Visualization of the rotated image
Min_dist = some arbitrary value > 0
for each pixel on the grid **do**

  Create a matrix, *distances_black*, which contains the Euclidean distances between the pixel on the grid and the coordinates of the black rotated pixels. Only fill a matrix entry when the distance is smaller than or equal to *Min_dist* and when the rotated black pixel at hand is closer to the pixel on the grid than any other rotated black pixel so far. Update *Min_dist* during the process if a smaller minimum distance is found.

    if *distances_black* is empty, **then**
      proceed with the next pixel on the grid,
    else
      assign black to the pixel on the grid.

Algorithm 1. Black-only rotation: *MEin_dist* visualization method.

The algorithm takes the coordinates of the rotated black pixels of the image as an input. Next, the value for *Min_dist* is set, which specifies the mentioned range. Then, for each pixel on the grid, the distance matrix is filled, which contains the distances between the pixel on the grid and the black pixels of the rotated image. Thus, it is checked for each pixel on the grid whether a rotated black pixel is within the range of *Min_dist*. Additionally, the distance is only stored if the black pixel at hand is closer than any other black pixel. Once the matrix is filled for a pixel on the grid, it contains the distances of the rotated black pixels to the pixel on the grid, provided that the distance is smaller than *Min_dist*. If this distance matrix is empty, no rotated black pixel is within reach and thus, the pixel on the grid should not be black but should stay empty. On the other hand, if the distance matrix contains at least one non-empty entry, it implies that at least one rotated black pixel is within reach. Therefore, the

corresponding pixel on the grid should be black. The algorithm terminates when each pixel on the grid is dealt with and, as a result, the grid of pixels represents the rotated image after visualization. Varying the value of *Min_dist* allows us to analyze and compare different sensitivity levels.

The *Min_dist* method relates to the development of the Game of Life since black pixels can die, be born or survive. A pixel has a square shape with height and width equal to 1 and a diagonal distance of $\sqrt{2}$. The search space is indicated by a dashed gray circle. Additionally, the gray dot refers to the center of a pixel and the black dots indicate centers of rotated black pixels. The following describes how the *Min_dist* method connects to the Game of Life.

Die. For all chosen values of *Min_dist*, it may happen that two or more pixels are within the search space since the pixels (both before and after rotation) have distance at least one from each other. Furthermore, assuming that the original image has at least two black pixels that are neighbors, there exist at least two pixels whose rotated coordinates have distance one. As an example, Figure 3 displays how two black pixels' coordinates can lie within the same pixel on the grid such that even though the pixel is assigned to black, two black pixels are transformed into one black pixel.

Figure 3. *Min_dist*: One of the two black pixels dies.

Be Born. Depending on the value chosen for *Min_dist*, search areas of neighboring pixels may overlap in one or more points. Even when the search areas only touch at one point, this point is considered as overlapping since the distance that is investigated is up to and including *Min_dist* units. In Figure 4, the black dot is part of both search areas. Assuming that no other black pixel's rotation coordinates are within either of the two search spaces, that black pixel would cause the two pixels on the grid to turn black. Thus, out of one black pixel in the input image, two black pixels emerge in the visualized rotated image. Hence, a new black pixel gets born.
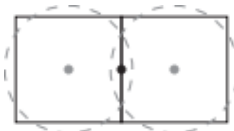

Figure 4. *Min_dist*: A new black pixel gets born.

Survive. A black pixel survives a rotation when a black pixel's rotated coordinates are within the search space of a pixel on the grid. Assuming no other black pixel's coordinates are within the search space, then that pixel solely accounts for its survival. An illustration is given in Figure 5.

Figure 5. *Min_dist*: A black pixel survives.

3.2.2 *Variation* Visualization Method for Zero-Lag Black-Only Rotation

Due to having to set a value for *Min_dist*, that approach is prone to create errors during visualization. The reason is that the image rotation outcomes may significantly depend on the parameter value of *Min_dist*. Furthermore, it could be controversial to stretch the search space in a circular shape since the pixels have a square shape. Therefore, the *Min_dist* method should be compared with the *Variation* visualization method, which searches within a square search space. Note that only the visualization procedure is adapted; the rotation procedure is the zero-lag black-only rotation as before.

Variation. Algorithm 2 describes the variation of Algorithm 1, which now does not employ a circular, but instead, a square search space. Thus, a pixel's cell on the grid is equivalent to the search space, and the center of the pixel is the center of the square search space. As before, the input of the algorithm is the coordinates of the rotated black pixels. Similar to the *Min_dist* method, the *Variation* algorithm iterates through every pixel on the grid to check whether it should be black or empty.

Input: Coordinates of each black pixel of the rotated image
Output: Visualization of the rotated image
for each pixel *a* on the grid **do**
  for each rotated pixel *b* **do**
    if the rotated pixel *b* is black, **then**
      if *b*'s rotated coordinates are within the cell of pixel *a* on the grid, **then**
        assign black to pixel *a* on the grid,
    else
      continue with next rotated pixel *b*.

Algorithm 2. Black-only rotation: *Variation* visualization method.

Note that Algorithm 2 takes each rotated black pixel and checks whether its rotated coordinates are within the pixel that is considered at the time. The cell boundaries are included in the area of a pixel. Thus, the pixel on the grid is colored black if a rotated black pixel is within the pixel on the grid (or its boundaries). Otherwise, the pixel on the grid stays empty. The algorithm iterates through all pixels on the grid and performs the described procedure. The output grid then resembles the visualized rotated image. Iterating through all black rotated pixels might seem inefficient to decide whether a pixel should be black since it is sufficient when at least one black rotated pixel is in the cell. However, a complete investigation allows us to see whether more than one black rotated pixel is in the cell.

One of the advantages of the *Variation* method is its independence from any parameter such as *Min_dist*, which reduces the susceptibility to errors. Similar to the *Min_dist* method, the *Variation* method relates to the Game of Life too, since black pixels can die, be born and survive. This is elaborated upon in the following.

Die. The minimum Euclidean distance between two pixels equals one, no matter whether they lie on the integer coordinates of the pixel grid or have been rotated. However, the diagonal of a pixel equals $\sqrt{2} \approx 1.4142$. This implies the potential occurrence of two black pixels having their rotated coordinates within one pixel, as displayed in Figure 6. Only one pixel is colored black during the visualization process, even though two black pixels are mapped into the pixel. Hence, two black pixels are combined into one and one of the black pixels dies during the visualization process.
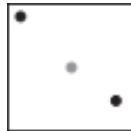

Figure 6. *Variation*: One of the two black pixels dies.

Be Born. When a black pixel is exactly mapped on the boundary between two pixels, it is considered within range for both pixels on the grid. Hence, both pixels would be colored black during the visualization process. Assuming there is no other black pixel whose rotated coordinates are within either of the two empty pixel cells, two black pixels arise out of one original black pixel. Thus, one pixel gets born, as shown in Figure 7.

Survive. A black pixel survives a rotation if it is mapped inside a pixel, including boundaries, and if it is the only pixel with its rotated coordinates in that specific pixel. Then, exactly one pixel is visualized

as black after exactly one black pixel has been rotated into that cell, as illustrated in Figure 8.
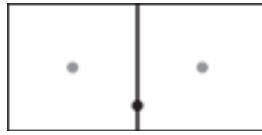

Figure 7. *Variation*: A new black pixel gets born.

Figure 8. *Variation*: A black pixel survives.

3.2.3 Visualization Method for Zero-Lag and One-Lag Black and White Rotation

The black and white rotation visualization is different from the two black-only methods (*Min_dist* and *Variation*) since there is more than one color to deal with. Within the black and white rotation, there are a zero-lag and a one-lag rotation method. These two methods only differ in what is taken as an input for the subsequent rotation. In the zero-lag rotation, the visualized image of the previous rotation is taken as an input, whereas in the one-lag rotation, the previous rotation's coordinates are used as an input.

The visualization procedure is the same for both types of rotation procedures. Recall that every pixel of the original image is colored; that is, there are no empty pixels. During the visualization procedure, each pixel on the grid checks which rotated pixel is closest. If there is one pixel with the minimum distance, the pixel on the grid takes on the color of the closest pixel. If there are two or more pixels with the minimum distance and the same color, the pixel on the grid takes on that color. If two or more pixels of different colors have the minimum distance to the pixel on the grid, the more frequent color of those pixels dominates. For example, if three rotated pixels have minimum distance from the pixel on the grid and two of them are white and one is black, then the pixel on the grid is colored white. In case two or more pixels with minimum distance compete and no color is (locally) more frequent than the rest, then black dominates and the pixel on the grid is assigned black. This assumes that one of the two colors in the image is black. Any color could be chosen as the dominant color. This visualization method ensures that each pixel in the visualized rotated image has an assigned color. An additional property of this method is

that it is easily extendable to more than two colors since the color is selected based on minimum distance and on a superior color rule. Algorithm 3 presents the visualization algorithm for the black and white rotation.

Input: Coordinates and color of each pixel of the rotated image
Output: Visualization of the rotated image
for each pixel a on the grid **do**

Find the rotated pixel(s) b_1, b_2, ... with minimum distance to pixel a on the grid.

  if there is only one rotated pixel with the minimum distance to the pixel on the grid, or if there are several with the same color, **then**

    assign the color of the pixel with the minimum distance to the pixel on the grid

  else

    if the pixels b_1, b_2, ... within the minimum distance have different colors and one color is more frequent than the other one, **then**

      assign the more frequent color to the pixel on the grid,

    else

assign black to the pixel on the grid.

Algorithm 3. Black and white rotation: Visualization method.

Next, we summarize how pixels can die, be born and survive in the black and white rotation visualization method to conform with the Game of Life.

Die. A black pixel can die during the process if it is outnumbered during a competition between pixels with different colors, as described earlier in this section. Moreover, a pixel can die at the corner pixels of an image. Take Figure 9 and assume it is the right-top corner pixel of an image. Even though the corner pixel is assigned to black, the outer pixel still dies. The reason is that it cannot be closest to any other pixel on the grid due to being in the corner of the image.
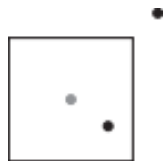

Figure 9. Black and white rotation: One of the two black pixels dies.

Be Born. Similar to the *Variation* method, a black pixel can be born when it is mapped on the boundary between two (or more) pixels. Assuming there is no other color to compete with, two black pixels emerge out of one black pixel, as shown in Figure 10.
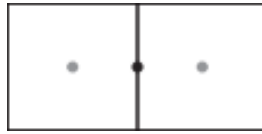
Figure 10. Black and white rotation: A new black pixel gets born.

Survive. A black pixel survives a rotation if it is the closest pixel to a pixel on the grid and no other pixel competes with it. Then, exactly one pixel is visualized as black after exactly one black pixel has been rotated into that cell, as illustrated in Figure 11.

Figure 11. Black and white rotation: A black pixel survives.

To summarize, the two black-only rotation visualization methods as well as the black and white rotation visualization method have the property to let pixels die, be born and survive. Thus, the original Game of Life definition is not violated.

▍ 3.3 Experimental Results

This section describes in what way the different methods from Section 3.2 are tested. Furthermore, the example images with which the methods get carried out are introduced.

3.3.1 The Rotation Methods

Black-only rotation. All black-only rotation experiments are performed using zero-lag rotation. Thus, the input for the next rotation iteration is the previous rotated visualized image. The *Min_dist* method is applied using four different values for the parameter *Min_dist*; 0.5, $\sqrt{1/\pi}$, $0.25(1 + \sqrt{2})$ and $0.5\sqrt{2}$, as presented in the following. The gray dashed circles indicate the corresponding search areas and the gray dots mark the centers of the pixels.

The first and the last case, Figures 12(a) and 12(d), represent the two extremes. The search area of the first case where *Min_dist* equals 0.5 contains the maximum area of the pixel while not crossing the pixel border. This search area is approximately 0.7854 square units big. Figure 12(d)'s search area includes the entire pixel, but therefore includes some area next to the pixel borders, too. The value of *Min_dist* is $0.5\sqrt{2}$ and the search area measures approximately 1.5708 square units.
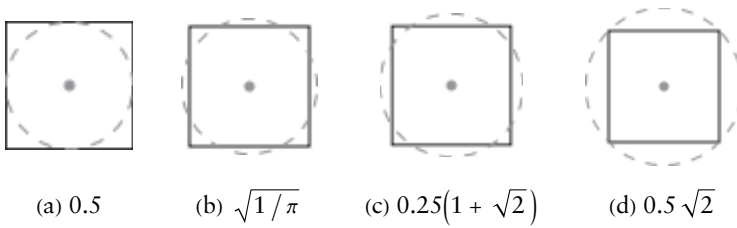
(a) 0.5   (b) $\sqrt{1/\pi}$ (c) $0.25(1 + \sqrt{2})$ (d) $0.5\sqrt{2}$

Figure 12. Overview of *Min_dist* and the search space.

As illustrated in the diagram, for the two values 0.5 and $0.5\sqrt{2}$ of *Min_dist*, both search areas with 0.7854 and 1.5708 do not even closely match the pixel's area of 1. Therefore, the average of those two search radii, so *Min_dist* equals $0.25(1 + \sqrt{2})$, is taken as a third parameter. This is illustrated in Figure 12(c). Even though additional space next to the pixel's borders is included, parts of the corners are excluded. The search area measures approximately 1.1444 square units and still overshoots the pixel's area by about 14%.

The fourth and last choice for *Min_dist* is $\sqrt{1/\pi}$, as shown in Figure 12(b). Here, the additional space outside the pixel exactly compensates the left-out corners. Thus, the search area equals 1.

Those four choices for *Min_dist* are applied within the experiments to evaluate the *Min_dist* method and to compare the different parameter settings (see Table 1). Furthermore, the given *Min_dist* parameters are used to compare this method to the *Variation* method, the second black-only rotation visualization method.

	Search Radius	Area of Search Space
Figure 12(a)	0.5	$0.25\pi \approx 0.7854$
Figure 12(b)	$\sqrt{1/\pi}$	1
Figure 12(c)	$0.25(1 + \sqrt{2})$	$\frac{\pi}{16}(3 + 2\sqrt{2}) \approx 1.1444$
Figure 12(d)	$0.5\sqrt{2}$	$0.5\pi \approx 1.5708$

Table 1. Overview of the radii and the search areas for different values of *Min_dist*.

The *Variation* method and the *Min_dist* method are applied to the same example images that get introduced in Section 3. Those two methods are compared with each other in Section 4, as they provide the two frameworks for the black-only rotation visualization.

Black and white rotation. The black and white rotation method is applied with the one-lag and the zero-lag rotation. Both approaches use the black and white rotation visualization method that was

introduced in Section 3. The results are compared and evaluated in Section 5.

3.3.2 The Images

The different rotation and visualization methods are tested using three example images called *Circle*, *Black Square* and *4 Squares*. The three images are squares and are horizontally, vertically and diagonally reflectional symmetric, as well as rotational symmetric, with degrees being multiples of 90. In the black-only rotation, the non-black pixels are treated as empty, whereas in the black and white rotation they are treated as white.

Furthermore, a more complex image, *Celtic Knot*, is investigated by the best-working rotation and visualization method (see Table 2). Note that *Celtic Knot* contains more pixels than the other example images and that it is not perfectly symmetric with respect to reflection. This image will not be considered in Sections 4 and 5.

	Circle	*Black Square*	*4 Squares*	*Celtic Knot*
dimension	25×25	33×33	21×21	63×63
number of black pixels	141	289	64	537
number of empty/white pixels	484	800	377	3432
proportion of black pixels	22.56%	26.54%	14.51%	13.53%
image				

Table 2. Overview of the example images.

3.4 Assessment of the Methods

This section introduces the measures that are used to assess the quality of the different methods and to compare them to each other.

3.4.1 Black Ratio

The first central measure to assess whether or not a rotation or a visualization procedure is satisfactory is to track the black ratio development as the percentage of the overall black pixels. Clearly, if the rotated image perfectly mimics the original image, they must share the same black ratio. On the other hand, a small deviation of the black ratio does not imply that the rotated image and the original image look similar. The reason is that the black ratio only considers the overall share of black pixels but not their locations on the pixel grid. Thus, two images can have the same black ratio but look completely different. Thus, the black ratio can be used to evaluate rotation methods, yet in a limited way.

3.4.2 Damage

The damage of a rotated image in comparison to the original image is defined as the relative deviation of pixel colors. For example, given an original image of 81 pixels and a rotated version of the image, the damage is obtained as follows: for each pixel it is checked whether the pixel color in the rotated image deviates from the color in the original image. If it does, it is added to the total number of errors. Then, the damage is calculated as

$$\frac{\text{total number of errors}}{(\text{total number of pixels} - 1)}.$$

Assuming the total number of errors in the example is 8, the damage is $8/80 = 10\%$. The denominator is reduced by one since the center pixel serves as the rotation center and is always mapped on itself. Thus, it cannot incur any damage. Moreover, to account for the image size and the total number of pixels, the relative and not the total deviation is considered.

Furthermore, the damage is only calculated and compared after a full rotation of 360 degrees. It is important to notice that it is not known whether a certain rotated pixel corresponds to exactly the same pixel in the original image since the intermediate pixel coordinates are not memorized. Thus, only the color of a certain pixel is compared.

3.5 Game of Life Applicability

For each rotation type, we will investigate the applicability to the Game of Life as a qualitative evaluation criterion in terms of its not being measurable as the other two criteria. In fact, it is left to the evaluation of observations such as the development of recognizable patterns, cycles and convergence patterns as well as the pixel development of being born, surviving and dying.

4. Black-Only Rotation Results

This section deals with the empirical results from the black-only rotation, and first, a few rotation examples are presented to illustrate the rotation procedure step by step. Then, different patterns that result from a full rotation are described. Next, the results from the black-only rotation are analyzed based on the three example images, which are described again in more detail.

4.1 Stepwise Rotation

To get a better understanding of the rotation procedure, the stepwise rotation is presented for a few examples in Figures 13 to 15. They

visualize the rotation processes of *Circle* around 40 degrees, of *Black Square* around 72 degrees and of *4 Squares* around 8 degrees, respectively, using different rotation methods.

The first image shows the original shape and the subsequent images show the successive clockwise rotation results around the respective angle. In Figure 15, the rotated images are first shown step by step and then transition into larger intervals of five rotation steps. For example, the image of Figure 15(h) corresponds to a total rotation of 80 degrees, and the next image, Figure 15(i), corresponds to a total rotation of 120 degrees.

(a) Original   (b) 1×40 degrees   (c) 2×40 degrees

(d) 3×40 degrees   (e) 4×40 degrees   (f) 5×40 degrees

(g) 6×40 degrees   (h) 7×40 degrees   (i) 8×40 degrees
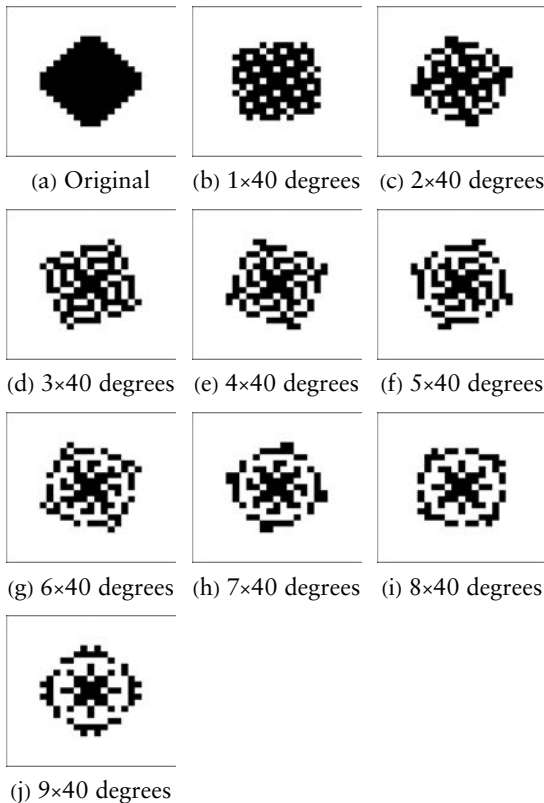
(j) 9×40 degrees

Figure 13. Step-by-step rotation: *Variation* method, *Circle*, 40 degrees.

Those image series show that the fully rotated image can look very different from the original image and that various patterns within as well as at the end of the rotation process are possible. Before the rotation outcomes are analyzed, common patterns that occur during the rotation are defined and labeled.
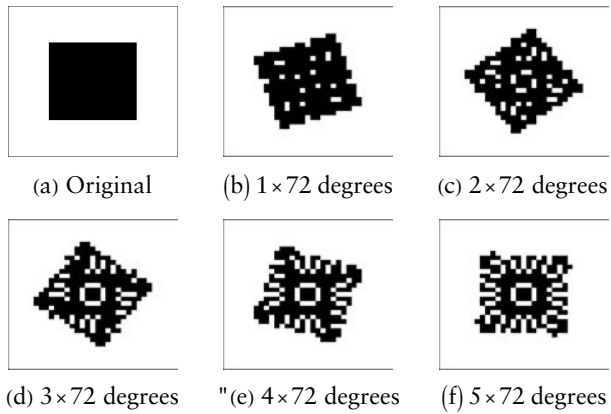
(a) Original   (b) 1×72 degrees   (c) 2×72 degrees

(d) 3×72 degrees   "(e) 4×72 degrees   (f) 5×72 degrees

Figure 14. Step-by-step rotation: *Min_dist* ($\sqrt{1/\pi}$) method, *Black Square*, 72 degrees.

(a) Original   (b) 1×8 degrees   (c) 2×8 degrees

(d) 3×8 degrees   (e) 4×8 degrees   (f) 5×8 degrees

(g) 6×8 degrees   (h) 10×8 degrees   (i) 15×8 degrees

(j) 20×8 degrees   (k) 25×8 degrees   (l) 30×8 degrees

Figure 15. (*continues*)
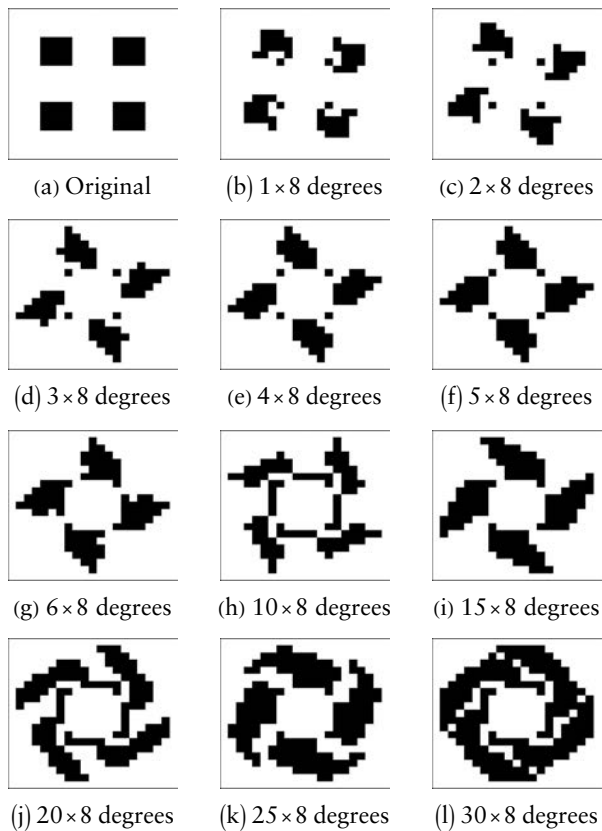
(m) 35×8 degrees   (n) 40×8 degrees   (o) 45×8 degrees
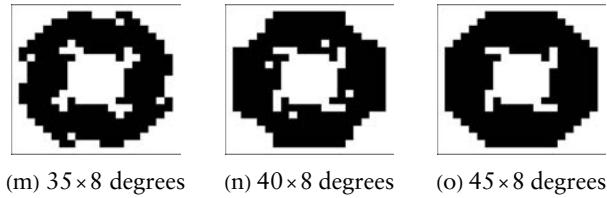
Figure 15. Step-by-step rotation: *Min_dist* $(0.25(1 + \sqrt{2}))$ method, *4 Squares*, 8 degrees

▌ 4.2 Different Patterns

One part of the analysis deals with the investigation and the comparison of the different shapes after a full rotation, so after $360/\alpha$ rotations around α degrees. In addition, it is analyzed whether the pattern stabilizes during the rotation process and whether there is a cycle of a few different patterns.

When two pixels touch each other diagonally, they are still defined as *disconnected* from each other since they do not share a pixel border. Two pixels are *strongly disconnected* from each other when they do not even touch each other diagonally. In the remainder, the shape of an image refers to its pattern after a full rotation. The two terms shape and pattern are used interchangeably. All examples in this section are taken from the black-only rotation. However, the shapes apply to the black and white rotation, too, which is treated in Section 5.

4.2.1 Original

In some cases, the image returns to its original after a full rotation. Clearly, this is a desirable outcome when it comes to minimizing the image rotation damage. Due to the successive alternation of rotation and visualization, this is not observed in many cases. A full rotation of *Circle* around 36 degrees using the *Variation* method yields the original image, as displayed in Figure 16(a).

4.2.2 Almost-Original

If the resulting image after a full rotation almost constitutes the original, it is labeled Almost-original. In the rotated example image, Figure 16(b), four black pixels are missing after a full rotation around 30 degrees of *4 Squares* by the *Variation* method.

4.2.3 Almost-Square

For some cases, the rotated image shows a figure that almost resembles square, as in Figure 16(c). This is the result of a full rotation of

Black Square around 45 degrees using the *Min_dist* method $(0.5\sqrt{2}\,)$. The image is missing some single pixels on the edge to be a proper square.
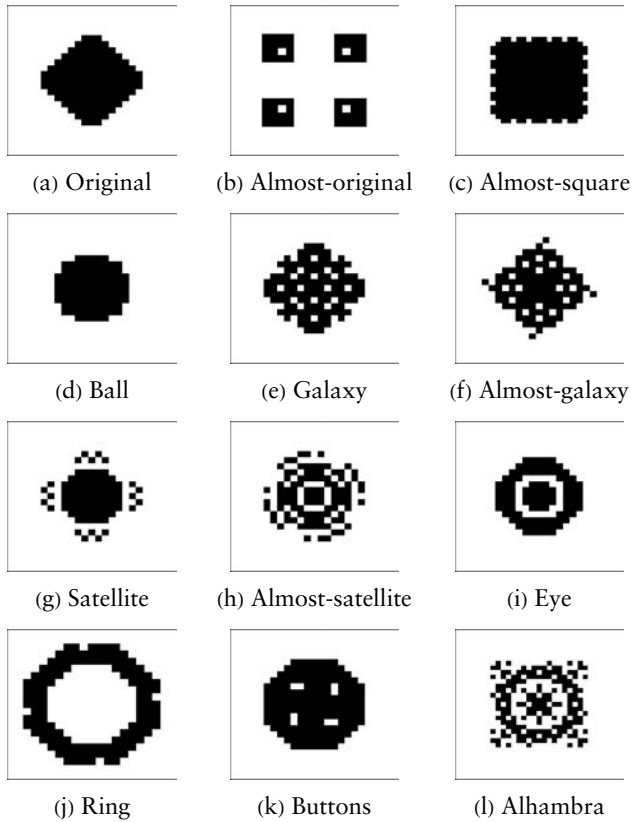

(a) Original	(b) Almost-original	(c) Almost-square
(d) Ball	(e) Galaxy	(f) Almost-galaxy
(g) Satellite	(h) Almost-satellite	(i) Eye
(j) Ring	(k) Buttons	(l) Alhambra

Figure 16. Overview of different patterns after a full rotation.

4.2.4 Ball

For an image to be classified as a ball, it is required to be of a circle-like shape with a solid interior, meaning that there are no holes in the image. Furthermore, it can neither have disconnected components nor tentacles, slim appendages of several pixels. For an example, refer to Figure 16(d), which is the resulting image of *Circle* after a full rotation around 5 degrees using the *Min_dist* (0.5) method.

4.2.5 Galaxy

The definition of a galaxy as a rotation result is very similar to the definition of a ball, with the only difference that the image is required to contain holes. The image is still characterized as circle-like and

without disconnected components or tentacles. Figure 16(e) shows such an image, which results after a full rotation around 45 degrees of *Circle* using the *Variation* method.

4.2.6 Almost-Galaxy

An almost-galaxy is defined as a galaxy, with the addition that it can have several disconnected components of exactly one pixel. Those disconnected components are required to touch the main body diagonally, as in Figure 16(f). Note the different classification of the images 16(e) and 16(f), which is solely due to the diagonally touching but disconnected single pixels in the second image. The image in Figure 16(f) is the fully rotated *Circle* around 60 degrees using the *Variation* method.

4.2.7 Satellite

A *satellite* has at least one strongly disconnected component. Note that one of the example images, *4 Squares*, is in itself a satellite since it has four strongly disconnected components. Another example of a satellite is rotating *Circle* around 36 degrees with the *Min_dist* (0.5) method, as displayed in Figure 16(g).

4.2.8 Almost-Satellite

An almost-satellite is very similar to a satellite, with the distinction that it has diagonally touching but disconnected components of at least two pixels. An example is provided in Figure 16(h), where *Circle* is fully rotated around 20 degrees with the *Min_dist* ($\sqrt{1/\pi}$) method. If an image has diagonally touching but disconnected components of only one pixel, it is no longer an almost-satellite but an almost-galaxy. Thus, those two classifications are related very closely.

4.2.9 Eye

An image is classified as an eye when it contains two strongly disconnected components, of which one of them fully encircles the other one. If at least one pixel of the inner component diagonally touches a pixel of the outer component, the image is no longer an eye but would probably be classified as a galaxy. For instance, rotating *Circle* fully around 12 degrees with *Min_dist* set to $\sqrt{1/\pi}$ results in an eye pattern, as displayed in Figure 16(i).

4.2.10 Ring

For an image to result in a ring, it needs to have an orbital pattern. This means that its interior must be empty and the ring itself must be completely connected such that there are no holes within the ring.

Figure 16(j) provides such a structure, which resulted from a full rotation of *4 Squares* around 9 degrees by the *Min_dist* ($\sqrt{1/\pi}$) method. The difference between a ring and an eye is that the former does not have an interior component.

4.2.11 Buttons

To be described as a button, an image is required to have a ball-like shape with exactly four empty or white non-diagonally touching shapes that can consist of more than one pixel each but must each consist of the same number of pixels. In a sense this definition almost equals the definition of a galaxy, however, here the number of holes is set to four. Nonetheless, every button image is a galaxy, too. During the analysis, the more restrictive classification is used. An example of buttons can be found in Figure 16(k), which is the result of a full rotation of *Circle* around 10 degrees with the *Min_dist* ($0.25(\sqrt{2}+1)$).

4.2.12 Alhambra

An Alhambra pattern has a less precise definition than all of the other patterns given. The images should resemble the geometric patterns of the Alhambra, a palace and fortress of the Moorish monarchs of Granada, Spain. For the analysis, there are no formal restrictions on connectedness nor shape of the image. It rather is a more qualitative classification. Additionally, the image is required to be "squiggly" and symmetric. An example of an Alhambra is the full rotation of *Black Square* around 40 degrees as depicted in Figure 16(l). That image is obtained by the *Variation* method.

4.2.13 Hybrid Patterns

In addition to the shapes described, there are hybrid patterns, too. For example, consider Figure 17, which could either be classified as an eye with some empty pixels in the outer component or as a galaxy. Such images might make the classification ambiguous; however, during the analysis, those cases are specified.

Figure 17. Example of a hybrid pattern.

▌ 4.3 Different Images

This section presents selected empirical results that include the developments of patterns and cycles as well as other characteristics in the

light of different rotation methods. For each of the three example images *Circle*, *Black Square* and *4 Squares*, five different rotation approaches are compared. This includes four versions of the *Min_dist* method as well as the *Variation* method. The comparison includes the behavior of the image series throughout a full rotation as well as the pattern of the fully rotated image. For each of the three example images, the rotation angles 8 and 12 degrees are selected, as well as one arbitrary angle.

For each image and for each selected angle, a table summarizes the results of the rotation (see Tables 3–5). The first row, "original," specifies whether, and if yes, at which iteration(s) the image returns to the original pattern. The next row specifies the cycling behavior, where the length of the cycle and its starting point are indicated. The third row indicates the stabilizing behavior, where the first rotation iteration of the stable state is listed. Rows four and five refer to the image after a full rotation. They specify whether or not the final image has disconnected components and how large the relative damage with respect to the original image is. Next, the final proportion of black pixels is given. See the table captions for a description of the development of the black pixel ratio and whether it increases or decreases and/or stabilizes. Finally, the shape of the fully rotated image is classified and the rotated image is displayed.

Some of the criteria (return to the original, cycles, stable state and black ratio development) refer to the series of images during the rotation process. Other criteria (disconnected components, damage and the shape) describe the image after a full rotation. Those comparison criteria are consistent for all three images and for both the black-only and the black and white rotation.

4.3.1 *Circle*

The comparison starts with the image *Circle* that is displayed in Figure 18. The image measures 25×25 pixels, of which 141 are black and 484 are empty, which implies a black proportion of 22.56%.

The first observation is that none of the rotated images ever returns to the original, not even during intermediate steps, until the full rotation. Even for all other rotation angles, *Circle* only returns to its original after a full rotation around 36 degrees by the *Variation* method. Obviously, the full rotations around 90, 180 and 360 degrees result in the original pattern, too.

Figure 18. *Circle*: Original image.

	Variation	0.5	$\sqrt{1/\pi}$	$0.25(1+\sqrt{2})$	$0.5\sqrt{2}$
original	-	-	-	-	-
cycle (length, start)	11, 17	11, 7	-	-	-
stable state (iteration)	-	-	-	-	39
disc. components	yes	yes	yes	-	-
damage	14.10%	16.03%	32.05%	26.92%	66.02%
black pixel ratio	16.16%	7.84%	39.20%	45.60%	88.48%
shape	eye	satellite	eye	buttons	ball
after 360 degrees					

Table 3. Black-only rotation: *Circle*, 8 degrees. Black pixel ratio: decreases, stabilizes; decreases, stabilizes; increases, stabilizes; increases, stabilizes; increases, stabilizes.

	Variation	0.5	$\sqrt{1/\pi}$	$0.25(1+\sqrt{2})$	$0.5\sqrt{2}$
original	-	-	-	-	-
cycle (length, start)	-	-	-	-	-
stable state (iteration)	8	5	5	7	-
disc. components	yes	yes	yes	yes	-
damage	5.13%	9.62%	5.77%	7.05%	59.62%
black pixel ratio	17.44%	12.96%	16.80%	23.20%	82.08%
shape	galaxy, eye	eye	eye	galaxy, eye	ball
after 360 degrees					

Table 4. Black-only rotation: *Circle*, 12 degrees. Black pixel ratio: decreases, stabilizes; decreases, stabilizes; decreases, stabilizes; increases, stabilizes; increases.

Considering the three selected angles, only the images rotated by the *Variation* method and the *Min_dist* (0.5) method cycle. Both cycles are of length 11. Furthermore, at 12 degrees, the series for all methods stabilize, except for the rotation with *Min_dist* equal to $0.5\sqrt{2}$. For 8 degrees, only the *Min_dist* ($0.5\sqrt{2}$) images reach a stable state. However, an investigation of the intermediate rotated images clarifies that the image only stabilizes due to the limitation by the grid. If the image grid were larger, it would probably not converge but expand further. Additionally, we can observe that the larger the value of *Min_dist*, the less likely it is to have disconnected components. This can be explained by the coloring by neighboring pixels since a larger value of *Min_dist* causes the search area to expand further beyond the pixel boundaries.

	Variation	0.5	$\sqrt{1/\pi}$	$0.25(1+\sqrt{2})$	$0.5\sqrt{2}$
original	-	-	-	-	-
cycle (length, start)	-	-	-	-	-
stable state (iteration)	-	-	-	-	-
disc. components	yes	yes	yes	yes	-
damage	12.18%	16.03%	12.82%	9.62%	28.85%
black pixel ratio	16.80%	10.40%	25.12%	33.88%	42.40%
shape	Alhambra	satellite, eye	galaxy	galaxy	ball
after 360 degrees					

Table 5. Black-only rotation: *Circle*, 24 degrees. Black pixel ratio: decreases, stabilizes; decreases, stabilizes; increases; increases.

Regarding the black ratio development, it seems that for smaller values of *Min_dist* the ratio decreases and for larger values it increases and that the threshold can vary. This is investigated further in Section 4.4. The rotation damage is treated in more detail in Section 4.5. The last observation is that the shape of a rotated image can vary depending on the method. The rotation around 12 degrees stands out since four of the five rotated images yield an eye shape. Since for two of the rotation outcomes the inner and the outer parts are connected, the patterns are strictly speaking not an eye anymore but a galaxy.

4.3.2 *Black Square*

The comparison continues with the image *Black Square*, which is displayed in Figure 19 and shows a black square of dimension 17 (see Tables 6–8). The image contains 1089 pixels in total (33×33), of which 289 are black and 800 are empty, implying a black proportion of 26.54%.

Figure 19. *Black Square*: Original image.

Black Square does not return to its original for any of the rotation angles (except for the multiples of 90 degrees). Moreover, only the image series of the 8-degree rotation by the *Variation* method and the image series of the 12-degree rotation by the Min_dist (0.5) method

	Variation	0.5	$\sqrt{1/\pi}$	$0.25(1+\sqrt{2})$	$0.5\sqrt{2}$
original	-	-	-	-	-
cycle (length, start)	11, 25	-	-	-	-
stable state (iteration)	-	7	13	-	
disc. components	yes	yes	yes	yes	-
damage	10.92%	19.12%	9.93%	12.87%	56.99%
black pixel ratio	17.72%	7.44%	23.23%	32.78%	83.48%
shape	satellite, eye	eye	eye	satellite, button	ball
after 360 degrees					

Table 6. Black-only rotation: *Black Square*, 8 degrees. Black pixel ratio: decreases, stabilizes; decreases, stabilizes; decreases, stabilizes; increases, stabilizes; increases.

	Variation	0.5	$\sqrt{1/\pi}$	$0.25(1+\sqrt{2})$	$0.5\sqrt{2}$
original	-	-	-	-	-
cycle (length, start)	-	7, 8	-	-	-
stable state (iteration)	-	-	-	-	
disc. components	yes	yes	yes	-	-
damage	12.13%	19.49%	12.87%	16.54%	48.16%
black pixel ratio	18.82%	8.54%	18.09% & 18.46%	33.52%	74.66%
shape	satellite, double eye	satellite, eye	satellite, double eye	galaxy, eye	ball
after 360 degrees					

Table 7. Black-only rotation: *Black Square*, 12 degrees. Black pixel ratio: decreases, stabilizes; decreases, stabilizes; decreases, alternates; increases; increases.

start to cycle. Only two series stabilize; the 8-degree rotation series generated by the *Min_dist* (0.5) method and by the *Min_dist* ($1/\sqrt{\pi}$) method. For 8 and 12 degrees, the rotations with *Min_dist* equal to $0.5\sqrt{2}$ are again subject to the physical limitation by the pixel grid. Furthermore, it is again observed that the larger the value of *Min_dist*, the less likely it is to have disconnected components. Again,

	Variation	0.5	$\sqrt{1/\pi}$	$0.25(1+\sqrt{2})$	$0.5\sqrt{2}$
original	-	-	-	-	-
cycle (length, start)	-	-	-	-	-
stable state (iteration)	-	-	-	-	
disc. components	yes	yes	-	-	-
damage	7.72%	18.75%	8.82%	4.78%	5.15%
black pixel ratio	21.76%	7.81%	18.46%	27.64%	31.68%
shape	galaxy, eye	almost-satellite, eye	galaxy	galaxy	almost-square
after 360 degrees					

Table 8. Black-only rotation: *Black Square*, 72 degrees. Black pixel ratio: decreases; decreases; decreases; increases; increases.

this can be explained by a large value of *Min_dist* enlarging the search area beyond the pixel boundaries. For the same reason, the black ratio increases rather than decreases for larger values of *Min_dist*. The rotation of 12 degrees and the value for *Min_dist* of $\sqrt{1/\pi}$ might provide the switching threshold between a decreasing and an increasing black ratio. Examining the black ratios of each rotation step of the $\sqrt{1/\pi}$ version around 12 degrees shows that the ratio alternates in a regular pattern between 18.09% and 18.46%. Even though both values are lower than the original image's black ratio, it does not settle at either value. Furthermore, no cyclic behavior in that image series can be observed. Regarding the damage, it can be pointed out that the rotation around 72 degrees with *Min_dist* of 0.5 performs exceptionally poorly, both comparing it with the damage within this method and comparing it across methods for the given degree and image. Finally, different shapes prevail after a full rotation.

4.3.3 *4 Squares*

The third and final image is *4 Squares*, as shown in Figure 20 (see Tables 9–11). This image contains four black squares of four times four pixels each. The four squares have the same distance from the center of the image. The pixel grid contains 441 cells (21×21), of which 64 are black and 377 are empty. The proportion of black pixels amounts to 14.51%.
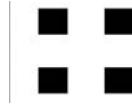
Figure 20. *4 Squares*: Original image.

	Variation	0.5	$\sqrt{1/\pi}$	$0.25(1+\sqrt{2})$	$0.5\sqrt{2}$
original	-	-	-	-	-
cycle (length, start)	-	11, 7	-	-	-
stable state (iteration)	-	-	-	-	26
disc. components	yes	yes	yes	-	-
damage	17.27%	14.55%	41.82%	44.55%	60.91%
black pixel ratio	9.98%	1.81%	47.17%	55.33%	75.28%
shape	satellite	satellite	galaxy	ring	ring
after 360 degrees					

Table 9. Black-only rotation: *4 Squares*, 8 degrees. Black pixel ratio: decreases, stabilizes; decreases, stabilizes; increases; increases; increases, stabilizes.

	Variation	0.5	$\sqrt{1/\pi}$	$0.25(1+\sqrt{2})$	$0.5\sqrt{2}$
original	-	-	-	-	-
cycle (length, start)	-	-	-	-	-
stable state (iteration)	-	-	-	-	22
disc. components	yes	yes	yes	-	-
damage	20.00%	17.27%	25.45%	35.43%	66.36%
black pixel ratio	9.07%	4.54%	21.77%	40.82%	80.73%
shape	satellite	satellite	satellite	galaxy	ring
after 360 degrees					

Table 10. Black-only rotation: *4 Squares*, 12 degrees. Black pixel ratio: decreases, stabilizes; decreases, stabilizes; increases; increases; increases, stabilizes.

	Variation	0.5	$\sqrt{1/\pi}$	$0.25(1+\sqrt{2})$	$0.5\sqrt{2}$
original	-	-	-	-	-
cycle (length, start)	-	-	-	-	-
stable state (iteration)	-	-	-	-	
disc. components	yes	yes	yes	yes	-
damage	8.18%	7.27%	7.27%	18.18%	36.36%
black pixel ratio	11.79%	9.07%	10.88%	23.58%	50.79%
shape	satellite	satellite	satellite	galaxy	galaxy
after 360 degrees					

Table 11. Black-only rotation: *4 Squares*, 40 degrees. Black pixel ratio: decreases; decreases, stabilizes; increases; increases; increases.

For the selected rotation angles, *4 Squares* does not return to the original image at all. However, it should be mentioned that for an angle of 45 degrees, the image returns to its original after a full rotation using the *Min_dist* method with parameters $\sqrt{1/\pi}$ and $0.25(1+\sqrt{2})$. Out of the 15 selected rotation series, only the rotation of *4 Squares* around 8 degrees using the *Min_dist* (0.5) method starts to cycle. Furthermore, only two of the images stabilize, namely the rotations of *Min_dist* equal to $0.5\sqrt{2}$ for angles of 8 and 12 degrees. However, both patterns are adjacent to the border of the pixel grid. It is likely that the shapes would expand further if they were not limited by the size of the pixel grid. Again, it is observed that the larger the value of *Min_dist*, the less likely the pattern is to have disconnected components, due to the expansion of the search area beyond the pixel boundaries. For the same reason, the black ratio increases rather than decreases for larger values of *Min_dist*. In addition, for those examples it holds true that the larger the value of *Min_dist*, the larger the black ratio, and the larger the damage after a full rotation. This does not hold true in general, as is further investigated in Sections 4.4 and 4.5 . Finally, very different shapes after a full rotation are observed.

▌ 4.4 Black Ratio Development

The proportion of black pixels of an image is defined as the ratio of the number of black pixels to the total number of pixels in the grid. The black ratio provides a measure of goodness with regard to a rotation method. If all pixels on an image are black, the ratio of black pixels is 1, or 100%, and if all pixels are white, the ratio is 0, or 0%.

This section analyzes the black ratio development based on the example images and their rotation angles chosen in Section 4.3. The corresponding plots can be found in Figure 21.

(a) *Circle*, 8 degrees   (b) *Circle*, 12 degrees   (c) *Circle*, 24 degrees

(d) *Black Square*,   (e) *Black Square*,   (f) *Black Square*,
8 degrees   12 degrees   72 degrees

(g) *4 Squares*, 8 degrees   (h) *4 Squares*,   (i) *4 Squares*,
12 degrees   40 degrees
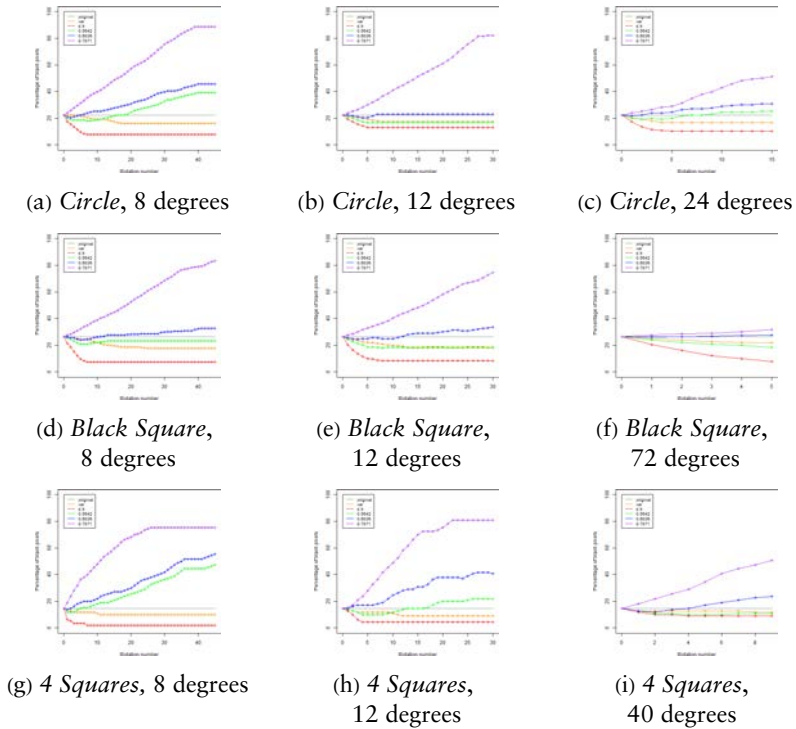
Figure 21. Black-only rotation: Overview of the black ratio deviation.

The *x* axis of each of the plots specifies the number of rotations needed for a full rotation, given a rotation angle. For instance, when a rotation around 8 degrees is performed, the *x* axis ranges from 0 to 45 since 45 rotations around 8 degrees add up to a full rotation of 360 degrees. The *y* axis reaches from 0 to 100, which corresponds to the proportion of black pixels in percent. In each of the plots, six different lines are presented, of which each corresponds to a discrete series of black ratios. The different discrete data points of the series are connected to allow an easier comparison between the different methods. The gray line serves as a benchmark and provides the black ratio of the original image. Next, the dark orange graph corresponds to the rotation of the example images by the *Variation* method. The other four graphs display the black ratio for the four methods of the *Min_dist* approach. For the different values of *Min_dist*, 0.5, $\sqrt{1/\pi}$,

$0.25(1 + \sqrt{2})$ and $0.5\sqrt{2}$, the colors red, green, blue and purple show the results, respectively.

The diagrams reveal that for the rotations of *Min_dist* (0.5), the black ratio is thoroughly and significantly lower than the original image's black ratio. On the contrary, for a *Min_dist* of $0.5\sqrt{2}$, the black ratio of the rotated image always overshoots the original black ratio. Note that in the plots for 72 and 40 degrees, the deviations are rather small, which can be explained by the short series of images (5 and 9, respectively). Thus, fewer rotation iterations imply that the black ratio has fewer opportunities to move away. As for the other three methods (*Variation* and *Min_dist* of $\sqrt{1/\pi}$ and $0.25(1 + \sqrt{2})$), each of them seems to be the best fit for some of the rotations, but it cannot be distinguished which method performs better than the others. Therefore, the black ratio deviations for the three example images and chosen angles are compared numerically and summarized in Table 12.

	Variation	0.5	$\sqrt{1/\pi}$	$0.25(1 + \sqrt{2})$	$0.5\sqrt{2}$
Circle					
8 degrees	29.67	201.93	87.49	197.40	1933.28
12 degrees	22.01	85.05	30.39	0.66	1278.85
24 degrees	28.81	132.27	4.59	29.03	318.31
Black Square					
8 degrees	54.11	339.07	13.10	10.69	1249.52
12 degrees	46.22	295.04	61.24	12.02	766.77
72 degrees	12.91	193.97	33.08	0.40	10.51
4 Squares					
8 degrees	17.50	153.61	355.83	598.33	2598.59
12 degrees	21.73	92.47	20.65	326.76	2536.03
40 degrees	4.19	25.11	14.52	25.43	529.37

Table 12. Black-only rotation: Average of squared deviation of black ratios.

Although a small black ratio deviation does not necessarily imply a good rotation in terms of little rotation damage, it gives an indication about how close the rotated image is to the original image. The values in Table 12 are the averages of the squared deviations of the black ratios and are calculated per method and per degree as

$$\frac{\alpha}{360} \sum_{i=1}^{360/\alpha} \left(\text{original black ratio} - \text{black ratio of } i^{\text{th}} \text{rotation of image}\right)^2$$

given a rotation around α degrees. Comparing the (averages of the) squares of the deviation of the black ratio penalizes further-away values relatively more since the deviations get squared. For example, if the deviation from the original image's black ratio is twice as high for rotation number 4 as for rotation number 10, then the penalty for deviation at rotation number 4 is four times as high as for rotation number 10 since the deviation gets squared.

Table 12 shows that choosing *Min_dist* to be $0.5\sqrt{2}$ performs notably worst in all cases except for the rotation of *Black Square* around 72 degrees. For this rotation, the *Min_dist* (0.5) method performs worst. The reason is that the image becomes very sparse, as shown in Table 8. Setting *Min_dist* to 0.5, $\sqrt{1/\pi}$ and $0.25(1+\sqrt{2})$ gives the least black ratio deviation in none, two and four cases, respectively. For the *Variation* method, there are three out of the nine cases for which the squared deviation is smallest. Thus, none of the methods can be suggested to perform best. However, for the *Min_dist* method for which the parameter equals 0.5 and $0.5\sqrt{2}$, the black ratio tends to deviate the most. Thus, the rotated images cannot be very similar to the original images. The analysis of the damage after a full rotation is performed for all rotation angles and for each of the example images in the following section.

▌ 4.5 Damage after a Full Rotation

This section analyzes and discusses the damage incurred after a full rotation for each of the three example images. All divisors of 360 are selected as rotation angles, and an assessment of the different methods is made after the damage analysis. As described in Section 3.4, the damage is defined as

$$\frac{\text{sum of errors}}{\text{total number of pixels} - 1} \tag{3}$$

where the sum of errors is defined as

$$\text{sum of errors} = \sum_{i=1}^{\text{dim}} \sum_{j=1}^{\text{dim}} \text{error} \tag{4}$$

$$\text{where} \begin{cases} \text{error} = 1 & \text{if img\_original}[i][j] \neq \text{img\_rotated}[i][j] \\ 0 & \text{otherwise} \end{cases}$$

and where dim is the number of pixel rows/columns of the image, img_original is the original image and img_rotated is the image that results from a full rotation. The denominator is reduced by one in equation (3) since the center pixel is always mapped exactly on itself. Therefore, it can neither be wrongly assigned nor incur any damage.

Figure 22 gives an overview of the relative damage in percent incurred in the three example images after a full rotation around each rotation angle by the different methods. On the x axis, the rotation degree is specified, which includes all divisors of 360. The dark orange line in each of the plots corresponds to the *Variation* rotation. The other four lines display the black ratio for the four versions of the *Min_dist* method. For the different values of *Min_dist*, 0.5, $\sqrt{1/\pi}$, $0.25(1+\sqrt{2})$ and $0.5\sqrt{2}$, the colors red, green, blue and purple display the results, respectively.
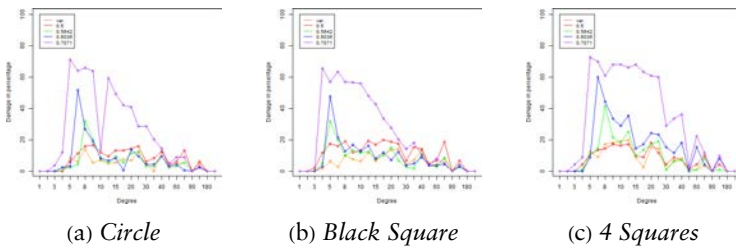

(a) *Circle*   (b) *Black Square*   (c) *4 Squares*

Figure 22. Black-only rotation: Damage after a full rotation.

The different rotation degrees on the x axis are evenly spaced to allow a better comparison of the specific rotation angles. Yet, it distorts the overall comparison slightly. The largest interval measures 180 degrees (between 180 and 360 degrees), but no conclusion can be deduced from the diagram for a rotation around any angle between 180 and 360 degrees. Since only the full rotation around angles that are divisors of 360 is investigated, this issue is nothing to be concerned about, yet it is important to keep in mind. Since the plots in Figure 22 compare the *damage* incurred after a full rotation, clearly minimal damage is preferred.

By only glancing at the plots, it already becomes clear that the rotation method with *Min_dist* of $0.5\sqrt{2}$ is performing worst in many cases. Furthermore, the plots suggest that all other methods perform better for some images and degrees and worse for others. To allow a better comparison, Table 13 gives a summary of the numerical results.

Depending on the size of the image, rotating around a very small angle does not change the rotated pixels' coordinates enough, which leaves the image unchanged by rotation. For the example images, this refers to the rotation angles of 1 and 2 degrees. Therefore, those cases are entirely left out of the analysis. Furthermore, Table 13 leaves out the cases 90, 180 and 360 degrees since those rotations can be performed perfectly on an orthogonal pixel grid. Thus, by definition,

they cannot incur any damage. This leaves 19 rotation angles, namely 3, 4, 5, 6, 8, 9, 10, 12, 15, 18, 20, 24, 30, 36, 40, 45, 60, 72 and 120 degrees.

	Variation	0.5	$\sqrt{1/\pi}$	$0.25(1 + \sqrt{2})$	$0.5\sqrt{2}$
Circle					
smallest angle having an impact	5	5	4	4	3
average damage	6.94%	11.05%	8.05%	10.08%	31.95%
minimum damage	0.00%	3.85%	1.28%	0.64%	3.21%
maximum damage	14.10%	16.67%	32.05%	51.95%	71.15%
♯ of times minimum damage	7 (35%)	0	7 (35%)	5 (25%)	1 (5%)
♯ of times maximum damage	1 (5%)	2 (10%)	0	0	17 (85%)
Black Square					
smallest angle having an impact	4	3	3	4	4
average damage	7.74%	13.68%	9.50%	11.42%	33.05%
minimum damage	1.84%	2.21%	0.37%	3.31%	4.04%
maximum damage	14.34%	20.22%	31.99%	47.79%	65.81%
♯ of times minimum damage	8 (40%)	0	8 (40%)	4 (20%)	0
♯ of times maximum damage	0	5 (26%)	0	0	14 (74%)
4 Squares					
smallest angle having an impact	5	4	4	5	3
average damage	9.52%	10.30%	12.12%	21.50%	43.16%
minimum damage	0.91%	1.82%	0.00%	0.00%	3.64%
maximum damage	20.00%	18.18%	41.82%	60.00%	72.73%
♯ of times minimum damage	8 (36%)	6 (27%)	6 (27%)	1 (5%)	1 (5%)
♯ of times maximum damage	0	0	0	0	19 (100%)

Table 13. Black-only rotation: Full rotation damage.

For each image and each method, the table summarizes the average, minimum and maximum damage in percentage. In addition, the number of times a method claims the minimum or maximum damage considering all rotation angles is listed. It can happen that the total number of times minimum damage is incurred (so the sum of the row

entries of "# of times minimum damage") is more than 19. The reason is that the same minimum rotation damage may be incurred by two or more methods.

Moreover, it is important to notice that the damage of a rotated image can be classified as both the minimum and the maximum damage of a certain degree. For instance, rotating *Circle* around 3 degrees only has an impact when *Min_dist* is set to $0.5\sqrt{2}$, but none of the other methods' rotations get affected. Hence, the damage created here is solely compared to itself. The same observation holds for rotating *4 Squares* around 3 degrees. Additionally, the image returns to its original after a full rotation three times: rotating *Circle* around 36 degrees using the *Variation* method and rotating *4 Squares* around 45 degrees using a *Min_dist* of either $\sqrt{1/\pi}$ or $0.25(1+\sqrt{2})$. For those three cases, the intermediate rotated images do not resemble the original image; only the final rotated image does.

As a comparison to get a better sense of the damage incurred, Table 14 gives an overview of the damage if either all pixels are black or all pixels are empty. This will happen if either the black pattern expands over the whole grid or if all black pixels disappear. In a way, that resembles the worst case damage since no inference on the original or the rotated image can be done. On the other hand, the maximum damage possible would be 100%, showing an inverse of the original image. Thus, we can distinguish between the maximum damage and the worst case damage.

	Damage in % if All Pixels Are Black	Damage in % if All Pixels Are Empty
Circle	77.56	22.44
Black Square	73.53	26.47
4 Squares	85.45	14.55

Table 14. Rotation damage: worst case.

Based on the results from the three example images, it can be derived that the rotation for *Min_dist* set to $0.5\sqrt{2}$ clearly performs worst. In fact, it accounts for 89%, 74% and 100% (or 17, 14 and 19 out of 19) of the rotated images with the maximum damage. The reason for this is, as described in Section 3.2, that whenever a pixel on the grid finds a rotated black pixel within (Euclidean) distance $0.5\sqrt{2}$, it turns black. This is equivalent to saying if there is a rotated black pixel within the circle of radius $0.5\sqrt{2}$ around the center of a pixel, this pixel is colored black. However, the corresponding circle

that is checked for rotated black pixels has an area of $\pi(0.5\sqrt{2})^2 \approx 1.5708$, whereas a pixel has an area of only 1. Hence, the search area extends well beyond the pixel area and expansion is unavoidable if *Min_dist* is set to $0.5\sqrt{2}$.

The other methods' performance varies among the example images. Overall, the *Variation* method, as well as setting *Min_dist* to $\sqrt{1/\pi} \approx 0.5642$, performs well. The reason is that often, a relatively low minimum damage and a relatively low maximum damage as well as a high number of attained minima and a low number of attained maxima are fulfilled. Note that in both cases the area of the search space equals 1, either as a square (*Variation*) or as a circle (*Min_dist* of $\sqrt{1/\pi}$). The performance of the remaining two methods, *Min_dist* set to 0.5 or $0.25(1+\sqrt{2})$, is average. For *Min_dist* set to 0.5, the minimum damage is relatively high, whereas the maximum damage is relatively low. The opposite is observed when *Min_dist* is set to $0.25(1+\sqrt{2})$.

If the methods are ranked based on their performance regarding the rotation damage, it looks as follows, 1 being the best and 5 being the worst:

1. *Variation*

2. *Min_dist* of $\sqrt{1/\pi}$

3. *Min_dist* of $0.25(1+\sqrt{2})$

4. *Min_dist* of 0.5

5. *Min_dist* of $0.5\sqrt{2}$

This section ends with a comparison of the damage and the change in the black ratio after a full rotation. Like previously, the analysis is based on three different rotation angles per image. For each image, each chosen angle and each method, the total number of changes in black pixels and the total number of errors are listed. The total number of changes in black pixels is the absolute difference between the number of black pixels in the original image and the number of black pixels in the rotated image. Furthermore, the total number of errors is the sum of the deviations regarding black versus empty pixels when comparing the original image and the rotated image, as defined in equation (4).

The far right column of Table 15 (*Min_dist* is $0.5\sqrt{2}$) suggests that the damage is solely due to the increase of black pixels, which—studying the resulting images—gains credibility. Additionally, for each degree and method, the total number of changes in black pixels

is smaller than or equal to the total number of errors. This makes perfect sense since the damage describes not only the blackness itself but also includes the location of the black pixels. Unfortunately, no conclusion such as "given a rotation angle, if the changes of blacks are higher, then the damage is higher, too" can be drawn. The reason is that the results are, for example, *total change black: 20/56/144, total # of errors: 76/64/184* for methods *Variation*, *Min_dist* of 0.5 and *Min_dist* of $\sqrt{1/\pi}$, respectively, when rotating *4 Squares* around 8 degrees. Thus, the scope of the damage cannot be inferred from the change in the number of black pixels only.

	Variation	0.5	$\sqrt{1/\pi}$	$0.25(1+\sqrt{2})$	$0.5\sqrt{2}$
Circle					
8 deg.: total change blacks	40	92	104	144	412
8 deg.: total ♯ of errors	88	100	200	168	412
12 deg.: total change blacks	32	60	36	4	372
12 deg.: total ♯ of errors	32	60	36	44	372
24 deg.: total change blacks	36	76	16	52	180
24 deg.: total ♯ of errors	76	100	80	60	180
Black Square					
8 deg.: total change blacks	96	208	36	68	620
8 deg.: total ♯ of errors	112	208	108	140	620
12 deg.: total change blacks	84	196	92	76	524
12 deg.: total ♯ of errors	132	212	140	180	524
72 deg.: total change blacks	52	204	88	12	56
72 deg.: total ♯ of errors	84	204	96	52	56
4 Squares					
8 deg.: total change blacks	20	56	144	180	268
8 deg.: total ♯ of errors	76	64	184	196	268
12 deg.: total change blacks	24	44	32	116	292
12 deg.: total ♯ of errors	88	76	112	156	292
40 deg.: total change blacks	12	24	16	39	160
40 deg.: total ♯ of errors	36	32	32	80	160

Table 15. Black-only rotation: Comparison of black ratio change and rotation damage.

▎ 4.6 Game of Life Applicability

The black-only rotation provides a version of John Conway's Game of Life with its own rules. The original Game of Life has a set of rules (Section 2.1) that determines which pixels are black and which ones

are white in the next generation. In this paper, a rotation operator is introduced as the game component. Starting from the well-known rotation matrix that is used to perform a rotation in Euclidean space, the game gets changed in the different versions of the visualization process. The general idea of the visualization is homogeneous: that each pixel in the grid checks whether or not any rotated black pixel is mapped into a pixel's search space. Depending on the method, the search space can be a circle with radius 0.5, $\sqrt{1/\pi}$, $0.25(1 + \sqrt{2})$ or $0.5\sqrt{2}$. Alternatively, the search space is a square of area 1 where the pixel center is the center of the square and the search space corresponds to the particular pixel on the grid.

Through successive rotations around different angles of α degrees, the game evolves over $360/\alpha$ rotations. The experimental results show examples of convergence to stable states. Hence, once a certain pattern is reached, rotating around the same angle does not change the image anymore. Furthermore, examples of circular behavior are found where the same sequence of images is obtained over and over again. Finally, some patterns evolve but do not converge or start to cycle within a full rotation, yet they might do so beyond 360 degrees. Especially for larger angles, this would be interesting to investigate since until now, only very few rotations have been considered. For example, for an angle of 72 degrees, only five rotations are performed. However, even for smaller angles such as 6 degrees, there are examples for which the pattern neither converges nor starts to cycle, but continues to evolve throughout the series of rotated images.

Relating the black-only rotation results to the Game of Life, the parallel between stabilization and still lives as well as between cycles and oscillators is unveiled. Those two concepts are in fact equivalent and are solely labeled differently. The cycle length is referred to as the period in the Game of Life. The third common pattern of the Game of Life, the spaceship, describes a finite pattern that returns to its original state after a finite number of iterations but is located elsewhere on the grid. However, a spaceship pattern is not observed in the rotations since, by definition, the pixel grid is limited by the size of the input image including the pixel grid. Furthermore, the rotation around the center pixel avoids the pattern to shift over to one side of the image.

It might be interesting to investigate the developments beyond a full rotation since in the real Game of Life many patterns eventually become a combination of still lives, oscillators and spaceships. Even though some patterns may seem (and some stay) chaotic, they can be chaotic for many iterations until they settle to one of the common patterns or a combination of them.

Finally, the conclusion is drawn that the black-only rotation with its different visualization techniques is classified as a novel variant of

the Game of Life. Depending on the initial image, an angle of α degrees and a visualization procedure, the patterns develop throughout the rotations. Some patterns reach a stable state; others converge or stay chaotic. Twelve different characteristic patterns are identified, and almost all rotated images can be classified as one of those patterns. The *Variation* method and the *Min_dist* ($\sqrt{1/\pi}$) method incurred the least rotation damage during the experiments. Regarding the visualization algorithm's efficiency, it is iterated through all pixels twice, namely through all pixels on the grid and through all rotated pixels. The rotated pixels are only considered if they are black, which speeds up the algorithm. This could be improved further by obtaining a subset of rotated pixels, based on the rotation angle, that are potential candidates to be assigned to a pixel on the grid. Nonetheless, no computational inefficiency problems are encountered during the experiments.

5. Black and White Rotation Results

This section continues with the analysis of the simulation results and investigates the rotation of black and white images. Here, the two colors black and white exist and each pixel is assigned exactly one of the two colors. Hence, empty pixels no longer exist and now every pixel is assigned a color during the visualization process. This feature is the main difference between the black-only and the black and white rotations. Furthermore, the black and white rotation splits into the zero-lag method and the one-lag rotation method. The patterns after a full rotation are classified according to the identified patterns in Section 4.2.

This section is structured similarly to the previous one. First, a few stepwise rotation examples for the zero-lag and the one-lag rotation methods are provided. Then, the impact of the rotation methods on the different images and the selected angles is analyzed. Afterward, the black ratio development of those images is investigated. Then, the rotation damage is examined and finally, the applicability to the Game of Life is assessed.

5.1 Stepwise Rotation

Next we illustrate a few rotations step by step. The images and the rotation degree are the same as in the stepwise examples in the previous section. Figures 23 to 25 show the rotation progress for *Circle* around 40 degrees, *Black Square* around 72 degrees and *4 Squares* around 8 degrees. *Circle* and *4 Squares* are rotated by the zero-lag black and white rotation method and *Black Square* is rotated by the one-lag black and white rotation method.
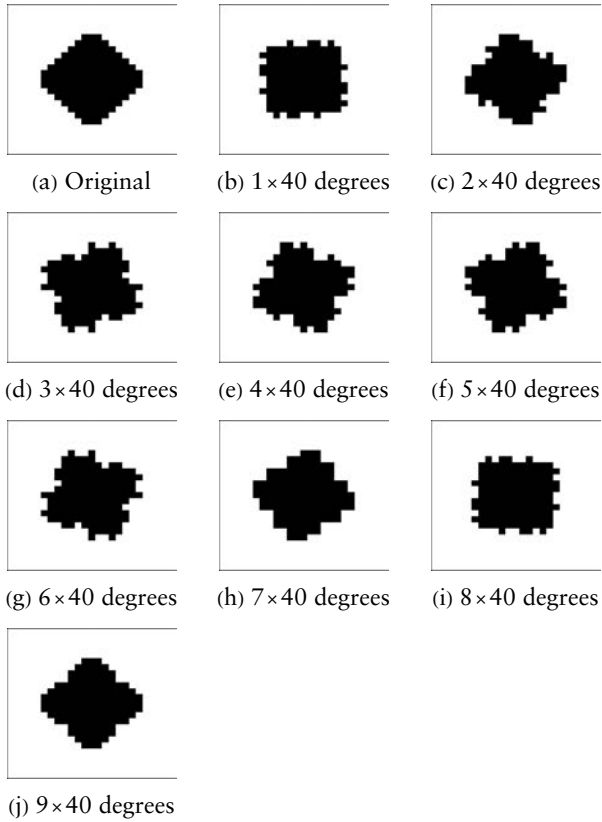
(a) Original   (b) 1×40 degrees   (c) 2×40 degrees

(d) 3×40 degrees   (e) 4×40 degrees   (f) 5×40 degrees

(g) 6×40 degrees   (h) 7×40 degrees   (i) 8×40 degrees

(j) 9×40 degrees

Figure 23. Step-by-step rotation: Zero-lag black and white rotation method, *Circle*, 40 degrees.

(a) Original   (b) 1×72 degrees   (c) 2×72 degrees

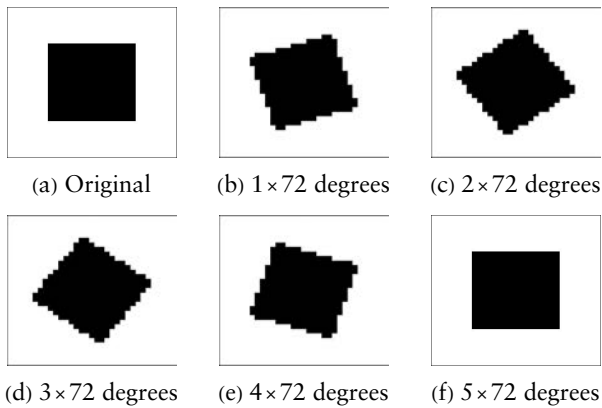(d) 3×72 degrees   (e) 4×72 degrees   (f) 5×72 degrees

Figure 24. Step-by-step rotation: One-lag black and white rotation method, *Black Square*, 72 degrees.

The first image of each series shows the original pattern, and the remaining images display the clockwise rotation around the given angle. Note that in Figure 25, the interval between the images jumps from one rotation iteration to five rotation iterations. The specific iteration number is denoted below the respective image.
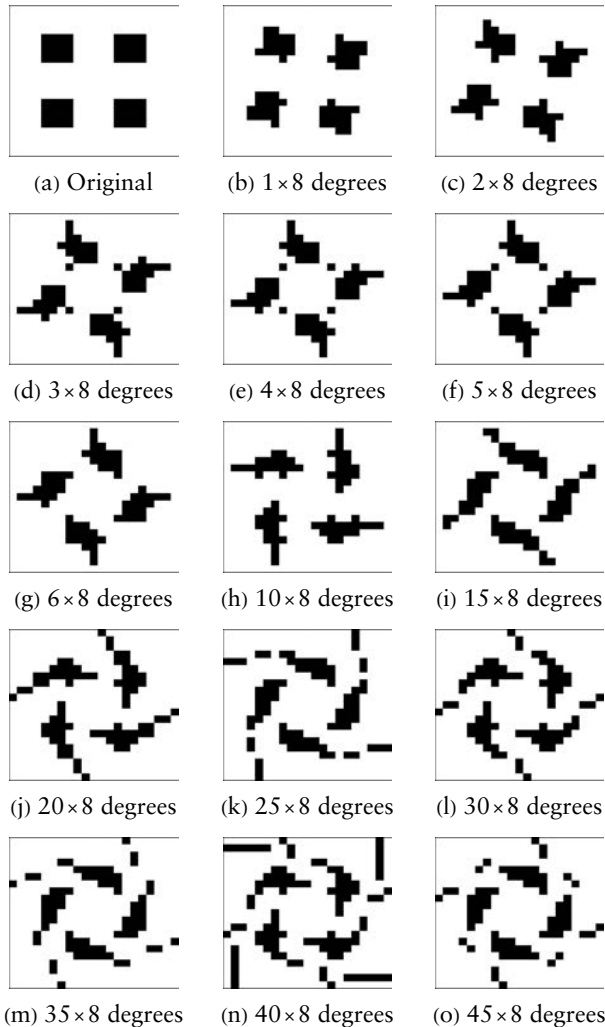

(a) Original   (b) 1×8 degrees   (c) 2×8 degrees

(d) 3×8 degrees   (e) 4×8 degrees   (f) 5×8 degrees

(g) 6×8 degrees   (h) 10×8 degrees   (i) 15×8 degrees

(j) 20×8 degrees   (k) 25×8 degrees   (l) 30×8 degrees

(m) 35×8 degrees   (n) 40×8 degrees   (o) 45×8 degrees

Figure 25. Step-by-step rotation: Zero-lag black and white rotation method, *4 Squares*, 8 degrees.

The zero-lag rotated *Circle* shows that the final pattern does not exactly mimic the original image, but it gets fairly close. The second example, the one-lag rotation of *Black Square*, however, returns to its

original pattern after a full rotation. Furthermore, the intermediate images look very similar to a square. Finally, the fully rotated image of *4 Squares* does not have much in common with the original pattern. It can be observed how the four small squares spread over the image and result in a symmetric final image. Here, the zero-lag rotation method does not preserve the initial pattern.

▍ 5.2 Different Images

The black and white rotation uses the same images and the same angles as in Section 4 to compare the zero-lag and the one-lag rotation methods with each other. The two methods differ with respect to the input of the rotation procedure. The zero-lag method takes the visualized image of the previous rotation as the input for the next rotation. The one-lag method uses the exact pixel coordinates of the previous rotated image as the input for the next rotation. An overview and a description of the three example images are provided in Table 2 and in Section 4.3. As before, the angles of interest are 8, 12 and 24 degrees for *Circle*; 8, 12 and 72 degrees for *Black Square* and 8, 12 and 40 degrees for *4 Squares*. The comparison criteria in the tables of this section are the same as for the black-only rotation.

5.2.1 *Circle*

As the rotation results in Table 16 suggest, the main difference between the zero-lag and the one-lag rotations is that the fully rotated image using the one-lag method always resembles the original image. This is due to memorizing intermediate rotated pixel coordinates for one iteration and using the exact values as an input for the next rotation step. Thus, even though intermediate images might look different from the original shape, the image always returns to the original pattern. Therefore, all image series that are generated by the one-lag rotation method are cyclic. For a rotation angle of α degrees, the cycle starts at the first image and has a length of at most $360/\alpha$ or a divisor thereof. Hence, if the image only returns to the original after a full rotation, the cycle length is $360/\alpha$. For example, a shorter cycle of length 18 is observed for the one-lag rotation of *Circle* around 5 degrees. In that case, the image returns to the original at rotations 18, 36, 54 and 72. Since the one-lag rotated images always return to their original after a full rotation, they do not show any rotation damage after a full rotation. Concerning the zero-lag rotation, it is observed that *Circle* does not return to its original at all. Furthermore, none of the selected cases starts to cycle. However, for 12 degrees, the image enters a stable state at rotation 13. The rotation damage and the black ratio are investigated further in Sections 5.3 and 5.4. Finally, different shapes are found after a full rotation by the zero-lag method.

	8 deg., 0-lag	8 deg., 1-lag	12 deg., 0-lag	12 deg., 1-lag	24 deg., 0-lag	24 deg., 1-lag
original	-	11, 22, 23, 34, 45	-	15, 30	-	15
cycle (length, start)	-	45, 1	-	15, 1	-	15, 1
stable state (iteration)	-	-	13	-	-	-
disc. components	yes	-	-	-	yes	-
damage	14.10%	0.00%	7.05%	0.00%	5.13%	0.00%
black ratio	18.72%	22.56%	29.60%	22.56%	21.28%	22.56%
shape	almost-satellite	original	ball	original	almost-galaxy	original
after 360 degrees						

Table 16. Black and white rotation: *Circle*.

5.2.2 *Black Square*

For *Black Square*, it is observed that only images rotated by the one-lag rotation method ever return to the original image (see Table 17). Furthermore, all one-lag rotated images cycle, with a cycle length of at most $360/\alpha$, where α degrees is the rotation angle. Additionally, the *Black Square* starts to cycle at rotation 27 when it is rotated around 8 degrees by the zero-lag black and white rotation method. It is observed that the black ratio for 8 and 12 degrees using the zero-lag rotation alternates between a few values. However, a dependency with the shape of the image cannot be verified. Finally, the zero-lag rotation method produces different patterns after a full rotation.

	8 deg., 0-lag	8 deg., 1-lag	12 deg., 0-lag	12 deg., 1-lag	72 deg., 0-lag	72 deg., 1-lag
original	-	11, 34, 45	-	15, 30	-	5
cycle (length, start)	11, 27	45, 1	-	15, 1	-	5, 1
stable state (iteration)	-	-	-	-	-	-
disc. components	-	-	yes	-	-	-
damage	8.82%	0.00%	5.15%	0.00%	2.21%	0.00%
black ratio	34.25%, 34.62% & 33.88%	26.54%	25.44% & 25.80%	26.54%	25.80%	26.54%
shape	ball	original	almost-galaxy	original	almost-square	almost-original
after 360 degrees						

Table 17. Black and white rotation: *Black Square*.

5.2.3 *4 Squares*

4 Squares only returns to the original when the image is rotated by the one-lag rotation method (see Table 18). Thus, those series with rotation angle of α degrees are cyclic of length at most $360/\alpha$. None of the image series rotated by the zero-lag method develops cycles or stabilizes. Compared to the other images, the damage incurred is rather high; however, the black ratio stays relatively close to the original black ratio of 14.51%. Due to the original image's nature, all rotated images have disconnected components. Therefore, the zero-lag rotated images are generally very likely to be classified as satellites since their original image is a satellite.

	8 deg., 0-lag	8 deg., 1-lag	12 deg., 0-lag	12 deg., 1-lag	40 deg., 0-lag	40 deg., 1-lag
original	-	11, 22, 23, 34, 45	-	15, 30	-	9
cycle (length, start)	-	45, 1	-	15, 1	-	9, 1
stable state (iteration)	-	-	-	-	-	-
disc. components	yes	yes	yes	yes	yes	yes
damage	24.55%	0.00%	21.52%	0.00%	8.18%	0.00%
black ratio	17.23%	14.51%	12.7%	14.51%	17.23%	14.51%
shape	satellite	original	satellite	original	satellite	original
after 360 degrees						

Table 18. Black and white rotation: *4 Squares*.

5.3 Black Ratio Development

This section treats the proportion of black pixels throughout rotation steps and analyzes the development graphically and numerically. The definition of the black proportion is the same as before; it is the ratio of the number of black pixels to the total number of pixels on the image. The black proportion can be anything between 0, or 0%, and 1, or 100%, if the image turns out all white or all black, respectively. Figure 26 displays the plots of the black ratio development for the chosen angles from Section 5.2. The x axis of each of the plots has as many units as rotation steps are needed for a full rotation. The y axis ranges from 10 to 35, referring to the black proportion (in percent). The y axis is adapted from the original range of 0 to 100 since the plotted black ratios only range from 11.79% to 34.62% and do not deviate heavily from the black ratio of the original image. Adjusting the range of the y axis allows a better graphical comparison of the methods. Each of the plots contains three lines, of which the gray line is constant and marks the original image's black ratio. The red and

blue points display the black ratio incurred at different rotation steps by the zero-lag and one-lag rotation methods, respectively.

(a) *Circle*, 8 degrees   (b) *Circle*, 12 degrees   (c) *Circle*, 24 degrees

(d) *Black Square*,   (e) *Black Square*,   (f) *Black Square*,
8 degrees   12 degrees   72 degrees

(g) *4 Squares*, 8 degrees   (h) *4 Squares*,   (i) *4 Squares*,
12 degrees   40 degrees
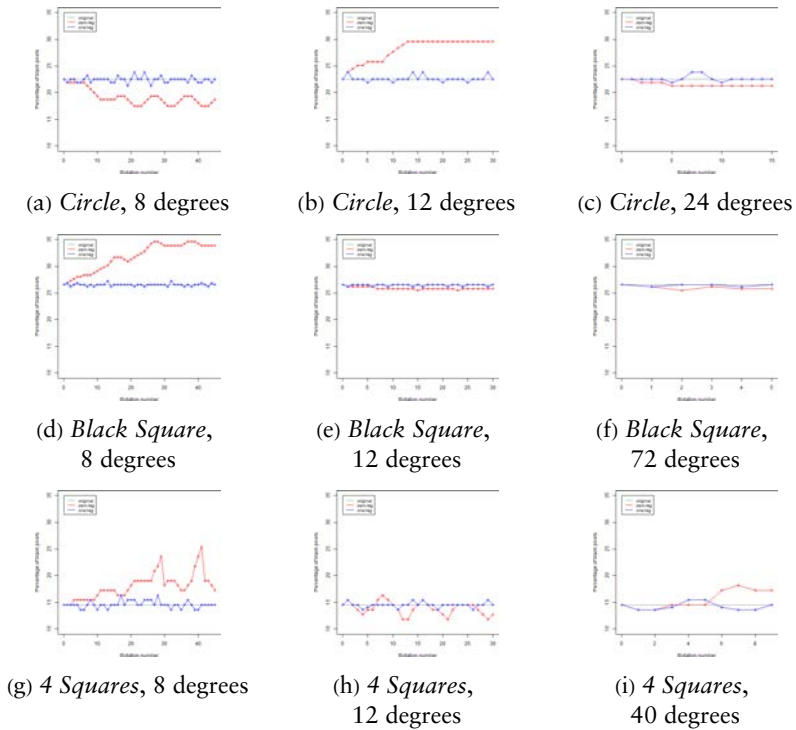
Figure 26. Black and white rotation: Overview of the black ratio deviation.

The graphical analysis shows that the black ratios of the images rotated by the one-lag rotation method are always closer to the original black ratio than the black ratios of the zero-lag rotated images are. Furthermore, the points referring to the zero-lag rotated images tend to be either strictly below or entirely above the one-lag points for the entire series of images. Thus, the two lines usually do not cross or meet. *4 Squares* provides an exception to this observation. The plot of the zero-lag rotation *Circle* around 8 degrees reveals a cycle of the black ratio from the fifteenth rotation onward. However, the underlying patterns do not show a cycling behavior.

Another observation is that for all black ratio plots of the one-lag rotated images, the first half of the plot is symmetric to the second half of the plot. For example, a full rotation around 8 degrees requires 45 rotations, and the plot shows a symmetry between the black ratio from rotation 0 to 22 and rotation 23 to 45. The accumulated rotation of 180 degrees corresponds to 22.5 rotations, which is

halfway between 22 and 23 degrees. Thus, we could hold a mirror at 22.5 to see that the two black ratios are symmetric. The same observation holds true for a symmetry between the first and the second quarter as well as between the third and the fourth quarter of the black ratio plot. This corresponds to rotation parts of 90 degrees each. Furthermore, the images themselves, for example, the first rotation and the second-last rotation, are mirrored. For instance, considering the 8-degree rotation, rotations 1 and 44 are symmetric, 2 and 43 are mirrored, et cetera. This makes perfect sense since the one-lag rotation always returns to the original image and for the one-lag rotation it does, in principle, not make a difference whether the image is turned clockwise or counterclockwise. The resulting images are exactly the same. That is, the one-lag counterclockwise rotation produces the same images as the one-lag clockwise rotation, just backward.

Moreover, after a full rotation, the zero-lag rotated images never have the original black ratio, whereas the one-lag rotated images always do. Thus, the one-lag rotated images seem to have a much better fit with the original image than the zero-lag rotated images when comparing the black ratio. A close fit of the black ratio is essential for a satisfactory rotated image that resembles the original image as closely as possible. Even though the one-lag rotated images always return to the original black ratio after a full rotation, the black ratio varies throughout the successive rotations. This implies that black pixels die and get born during the process.

A similar observation can be made from the black ratio deviations from the original black ratio. The averages of the squared black ratio deviations for the three example images and the chosen angles are compared numerically and summarized in Table 19. The values are calculated by

$$\frac{\alpha}{360} \sum_{i=1}^{360/\alpha} \left(\text{original black ratio} - \text{black ratio of } i^{\text{th}}\text{rotation of image}\right)^2$$

for each of the methods and each of the rotation angles of α degrees. A small deviation of the black ratio does not by itself imply that the rotated image is close to its original since it does not imply anything about the pattern of the image. However, a good fit between a rotated image and the original image needs the two black ratios to be close together. Thus, it is desirable that average deviations of the black ratios from the original black ratio be as small as possible.

The results show that the one-lag rotation method outperforms the zero-lag method since the average squared deviations are significantly lower for the rotations around each of the chosen rotation angles. The values in Table 19 take the squared deviation as a criterion. However, the conclusion does not change when the average absolute

deviation is taken instead. As already mentioned, a close black ratio does not necessarily imply a good rotation in terms of the rotation damage. Furthermore, only three rotation angles are investigated. Therefore, the rotation damage for all 24 divisors of 360 as rotation angles is analyzed in the next section.

	Zero-lag	One-lag
Circle		
8 degrees	14.12	0.29
12 degrees	35.73	0.27
24 degrees	1.28	0.27
Black Square		
8 degrees	34.67	0.07
12 degrees	0.51	0.05
72 degrees	0.52	0.04
4 Squares		
8 degrees	18.39	0.44
12 degrees	1.89	0.23
40 degrees	4.11	0.59

Table 19. Black and white rotation: Average of squared deviation of black ratios.

5.4 Damage after a Full Rotation

The next step of the analysis is the comparison of the zero-lag method and the one-lag method with respect to the damage after a full rotation. All divisors of 360 are considered as rotation angles, and the damage is defined by equations (3) and (4) in the previous section. The analysis for the black and white rotation damage follows the same structure as for the black-only rotation.

Figure 27 gives an overview of the relative rotation damage in percent after a full rotation for the three example methods and for each rotation angle. The x axis specifies the 24 rotation angles of interest. Note that the ticks are evenly spaced, yet they only provide discrete points. On the y axis, the damage in percent is denoted. The damage ranges from 0% to 30%, which is sufficient to display all data points. The red points visualize the damage incurred by the zero-lag rotation and the blue points describe the rotation damage using the one-lag rotation method.

The plots in Figure 27 show that a fully rotated image by the one-lag rotation method is never damaged. Thus, the visualized fully rotated image always resembles the original image. On the other hand, the example images that are rotated by the zero-lag method

show damage up to 24.55%. Even though this observation already strongly suggests that the one-lag method performs the image rotation in a perfect way, the two methods are still compared numerically. The comparison starts with a few statistics (average, minimum and maximum damage) in Table 20. The average damage for the zero-lag method takes only those angles into account for which the image is visually affected by the rotation. Thus, some of the small angles as well as 90, 180 and 360 degrees are left out.

(a) *Circle*   (b) *Black Square*   (c) *4 Squares*
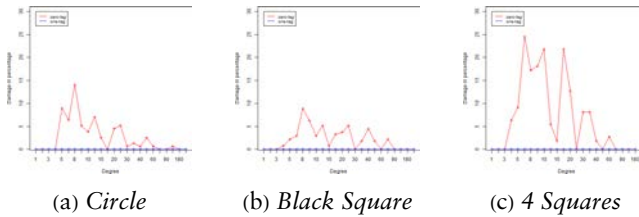
Figure 27. Black and white rotation: Damage after a full rotation.

	Zero-lag	One-lag
Circle		
smallest angle having an impact	5	1
average damage	3.77%	0.00%
minimum damage	0.00%	0.00%
maximum damage	14.10%	0.00%
Black Square		
smallest angle having an impact	4	1
average damage	2.90%	0.00%
minimum damage	0.00%	0.00%
maximum damage	8.82%	0.00%
4 Squares		
smallest angle having an impact	5	1
average damage	9.41%	0.00%
minimum damage	0.00%	0.00%
maximum damage	24.55%	0.00%

Table 20. Black and white rotation: Full rotation damage.

Furthermore, the two measures "number of times being minimum damage" and "number of times being maximum damage" from the black-only rotation analysis are left out. The reason is that whenever

the zero-lag rotation incurs any damage, it is automatically worse than the one-lag rotation method. In other words, the zero-lag rotation method never performs better than the one-lag rotation method with respect to the rotation damage. When the rotation damage is zero, it cannot be inferred that a specific pixel of the original image returns to exactly the same location in the rotated image. In other words, one cannot trace a pixel's location through the rotation steps. This is not essential for the analysis but important to keep in mind.

Regarding the zero-lag rotation method, all images achieve a minimum damage of 0% for some rotation angles. The rotation angles for *Circle* that are considered in Table 20 are 5, 6, 8, 9, 10, 12, 15, 18, 20, 24, 30, 36, 40, 45, 60, 72 and 120 degrees. Angles 1 to 4 are too small to change the image during the rotation. Furthermore, 90, 180 and 360 degrees are left out due to the perfect one-to-one mapping on the grid. Nonetheless, *Circle* returns to its original after a full rotation around 18 and 72 degrees. *Black Square* and *4 Squares* return to their originals for angles of 30, 60 and 120 degrees.

Overall, even the maximum damage incurred by the zero-lag rotation method is rather small, at least for *Circle* and *Black Square*, namely 14.40% (88 out of 625 pixels) and 8.82% (96 out of 1089 pixels). *4 Squares*, however, has a rather high maximum damage of 24.55% (108 out of 441 pixels). For five out of the 17 chosen angles, the damage of rotating *4 Squares* by the zero-lag method is more than 15%. This example shows that even though the zero-lag method seems to perform well with respect to the rotation damage when considering *Circle* and *Black Square*, it does not satisfy the expectations for *4 Squares*. Thus, the pattern of the image seems to influence the rotation performance with respect to the rotation damage.

Based on the results of the rotation damage, the two methods are ranked from best to worst as follows:

1. One-lag black and white rotation method
2. Zero-lag black and white rotation method

Finally, the black ratio development and the rotation damage are compared. Table 21 shows the total change of black pixels and the total number of errors (with respect to color) between the original image and the fully rotated image. The one-lag method performs flawlessly, whereas positive black deviations and errors are recorded for all instances of the zero-lag rotation method. However, no statement such as "the smaller/larger the total change of black pixels, the smaller/larger the total number of errors," or the other way round, can be drawn from the numbers at hand.

	Zero-lag	One-lag
Circle		
8 deg.: total change blacks	24	0
8 deg.: total ♯ of errors	88	0
12 deg.: total change blacks	44	0
12 deg.: total ♯ of errors	44	0
24 deg.: total change blacks	8	0
24 deg.: total ♯ of errors	32	0
Black Square		
8 deg.: total change blacks	80	0
8 deg.: total ♯ of errors	96	0
12 deg.: total change blacks	8	0
12 deg.: total ♯ of errors	56	0
72 deg.: total change blacks	8	0
72 deg.: total ♯ of errors	24	0
4 Squares		
8 deg.: total change blacks	12	0
8 deg.: total ♯ of errors	108	0
12 deg.: total change blacks	8	0
12 deg.: total ♯ of errors	96	0
40 deg.: total change blacks	12	0
40 deg.: total ♯ of errors	36	0

Table 21. Black and white rotation: Comparison of black ratio change and rotation damage.

▌ 5.5 Game of Life Applicability

The black and white rotation introduces a new variant of John Conway's Game of Life with different rules. The local operator of the original game is substituted by a rotation method that rotates two-color images. The black and white rotation method is, in general, not limited to two colors. The rotation process consists of two parts: the rotation itself and the visualization of the rotated image.

The two-color rotation has two different rotation approaches; the zero-lag method and the one-lag rotation method. The difference lies in the input for the rotation procedure. The zero-lag rotation method takes the previous visualized rotated image as the input. The one-lag rotation method uses the exact coordinates of the previous rotated pixels as the input. For both methods, the series of images evolves through sequential rotations around a fixed angle.

The analysis allows us to conclude that the zero-lag black and white rotation method can be classified as a new version of the Game of Life. The experiments with the zero-lag rotation method reveal

examples of convergence, cycles and chaotic patterns. The only common Game of Life pattern that cannot be observed is the spaceship, a finite pattern that returns to its original state after a finite number of iterations but at a different location. The reason is that the center pixel is fixed as the center of the rotation and this does not allow the image to shift over to one side. For the noncycling and nonstabilizing image patterns, it would interesting to investigate whether either of the two behaviors shows during further rotation steps beyond 360 degrees. This is left to future research.

Regarding the one-lag rotation, however, all patterns are cyclic and never converge or are chaotic. Given those observations of the patterns and relating them to the Game of Life, it can be concluded that the one-lag method behaves one-sided in the sense that all patterns are described as cycles. Even though the black ratio varies throughout the rotation, implying that black pixels die and get born, the long-run behavior of the image rotation with respect to cycling becomes predictable. Thus, it is not considered a qualified version of the Game of Life since the predictability lets the game character get lost.

To summarize, the one-lag method of the black and white rotation is not considered a true variant of the Game of Life since it always cycles. The series of rotated images never stabilizes or creates any chaotic patterns, which makes it rather predictable how the series develops. This lets the game character get lost. On the other hand, the zero-lag method of the black and white rotation is considered a variant of the Game of Life since the series of images can cycle, stabilize or be chaotic. Depending on the image and the rotation angle, the patterns develop differently throughout the rotation process. Furthermore, the pattern of the original image seems to have an impact on how badly the image is damaged after a full rotation. The black and white rotation visualization methods could be made more efficient by obtaining a subset of the pixels that are potential candidates to be mapped on a specific pixel on the grid. No computational inefficiency problems are encountered during the experiments.

6. Conclusion

This paper introduces a novel variant of the Game of Life that interlinks image rotation. The color of a cell is no longer determined by a function on its eight neighbors, but by the rotation angle and the color(s) of the pixel(s) that get rotated onto that cell. As in the Game of Life, the color refers to the state of a cell and the Game of Life rules are substituted with rotation and visualization procedures. Within this paper, two rotation methods and three visualization methods are

defined formally, tested and evaluated. The development of recognizable patterns as well as the long-run behavior of patterns, such as cycles and stable states, is studied in detail.

All black-only rotation methods create cycles and stable patterns, and most of the fully rotated images can be classified as one of the 12 identified recognizable patterns. Out of the black-only rotation methods, *Variation* and *Min_dist* ($\sqrt{1/\pi}$) perform better than the other three *Min_dist* versions with respect to the rotation damage. Thus, the value of *Min_dist* highly influences the rotation damage. The black ratio development does not prove itself to be a valid measure of the goodness of the rotation with respect to the similarity between the original image and the rotated image.

As a further result, the zero-lag black and white rotation method creates characteristic patterns, cycles and stable states. Moreover, the rotation damage after a full rotation is relatively low compared with the black-only rotation. On the other hand, all the one-lag black and white rotation method does is create cycles. Even though intermediate images might look very different from the original pattern, the image always returns to its original after a full rotation. The method works perfectly for the sole purpose of image rotation, but is not considered a valid application of the Game of Life due to its predictability.

The black-only *Variation* method creates cycles and chaotic patterns when it is applied to a larger and more sophisticated input image of a Celtic knot. Even though no stable states are observed within a full rotation, the patterns might well converge beyond a full rotation. Among all rotation experiments of the black-only rotation and the black and white rotation, a chaotic behavior of the rotated patterns is more common than cycles or stable states, but that is how it is in the real Game of Life.

Extending the rotations beyond 360 degrees would be a next logical step. The images might eventually start to cycle or settle at a stable state. Furthermore, we could consider larger images and extend the grid for the larger values of *Min_dist*. Here, the grid can be restrictive since the patterns may expand away from the center pixel. Alternatively, for smaller values of *Min_dist*, a bounding box framework could be introduced. This would specify the maximum size of the pattern, given an initial pattern and a value of *Min_dist*. Computationally speaking, all methods could be made more efficient by obtaining a subset of potential pixel candidates for each pixel during the visualization procedure. This would allow a subset of the rotated pixels to be considered as candidates for a specific pixel on the grid. Speeding up the algorithms becomes more crucial when the image size grows further.

Further research ideas may include more than two colors. For example, the described framework could be extended to a three-color

game in order to investigate rotation behavior and patterns. Finally, the rotation extension of the Game of Life could be implemented into other algorithms as a powerful mutation tool, as for example in the research about the Lanna character recognition [10]. Within a genetic algorithm, a full image rotation could be employed to generate the next mutation, provided that the full rotation changes the image.

In conclusion, the one-lag black and white rotation method does not qualify as a variant of the Game of Life due to its predictability, but it classifies as a good image rotation method. Furthermore, both the zero-lag black-only rotation method with its two visualization methods *Min_dist* and *Variation* and the zero-lag black and white rotation method provide novel variants of the Game of Life.

Acknowledgments

References

[1] M. Gardner, "Mathematical Games: The Fantastic Combinations of John H. Conway's New Solitaire Game 'Life'," *Scientific American*, **223**(4), 1970 pp. 120–123. www.jstor.org/stable/24927642.

[2] D. Eppstein, "Growth and Decay in Life-like Cellular Automata," *Game of Life Cellular Automata* (A. Adamatzky, ed.), London: Springer, 2010 pp. 71–97. doi:10.1007/978-1-84996-217-9_6.

[3] C. Bays, "Candidates for the Game of Life in Three Dimensions," *Complex Systems*, **3**(1), 1987 pp. 373–400. complex-systems.com/pdf/01-3-1.pdf.

[4] C. Bays, "A New Game of Three-Dimensional Life," *Complex Systems*, **5**(1), 1991 pp. 15–18. complex-systems.com/pdf/05-1-2.pdf.

[5] C. Bays, "Further Notes on the Game of Three-Dimensional Life," *Complex Systems*, **8**(1), 1994 pp. 67–73. complex-systems.com/pdf/08-1-4.pdf.

[6] C. Bays, "A Note on the Game of Life in Hexagonal and Pentagonal Tessellations," *Complex Systems*, **15**, 2005 pp. 245–252. complex-systems.com/pdf/15-3-4.pdf.

[7] C. Bays, "Cellular Automata in Triangular, Pentagonal and Hexagonal Tessellations," *Computational Complexity* (R. A. Meyers, ed.), New York: Springer, 2009 pp. 434–442. doi:10.1007/978-1-4614-1800-9_28.

 [8] M. Levene and G. Roussos, "A Two-Player Game of Life," *International Journal of Modern Physics C*, **14**(2), 2003 pp. 195–201. doi:10.1142/S0129183103004346.

 [9] M. R. Lauer, P. A. Mitchem and R. A. Gagliano, "Resource Optimization and Self Interest: Variations on the Game of Life," in *Proceedings of Simulation Symposium*, Phoenix, AZ, Piscataway, NJ: IEEE, 1995 pp. 136–143. doi:10.1109/SIMSYM.1995.393586.

[10] K. Khankasikam, "A Combined Genetic Algorithm and Conway's Game of Life for Printed Lanna Character Recognition," *International Journal of Computer Theory and Engineering*, **5**(4), 2013 pp. 653–657. doi:10.7763/IJCTE.2013.V5.769.

[11] G. Aguilera-Venegas, J. L. Galán-García, R. Egea-Guerrero, M. Á. Galán-García, P. Rodríguez-Cielos, Y. Padilla-Domínguez and M. Galán-Luque, "A Probabilistic Extension to Conway's Game of Life," *Advances in Computational Mathematics*, **45**(4), 2019 pp. 2111–2121. doi:10.1007/s10444-019-09696-8.

[12] S.-Y. Huang, X.-W. Zou, Z.-J. Tan and Z.-Z. Jin, "Network-Induced Nonequilibrium Phase Transition in the 'Game of Life'," *Physical Review E*, **67**(2), 2003 026107. doi:10.1103/PhysRevE.67.026107.

[13] M. Ibrahimi, O. Gulseren and S. Jahangirov, "Deterministic Phase Transitions and Self-Organization in Logistic Cellular Automata," *Physical Review E*, **100**(4), 2019 042216. doi:10.1103/PhysRevE.100.042216.

[14] S. M. Reia and O. Kinouchi, "Conway's Game of Life Is a Near-Critical Metastable State in the Multiverse of Cellular Automata," *Physical Review E*, **89**(5), 2014 052123. doi:10.1103/PhysRevE.89.052123.