6th International Congress on Scientific Research August 18-20. 2023, Ankara by IKSAD Institute

**New possibilities in the pixel-based cryptography**

Dr. László Pitlik (<https://orcid.org/0000-0001-5819-0319>),

László Pitlik (Jr.) (<https://orcid.org/0000-0002-8058-9577>)

Mátyás Pitlik (<https://orcid.org/0000-0002-1991-3008>),

e-Mails: [pitlik@my-x.hu](mailto:pitlik@my-x.hu), [ptlklszl@my-x.hu](mailto:ptlklszl@my-x.hu), [ppk@my-x.hu](mailto:ppk@my-x.hu)

Kodolányi János University and MY-X research team Hungary

*Keywords: cellular automata, genetic algorithms, binary/numeric pixel-value transformations, binary Caesar-code-like ciphers, de/coding of pictures (hand-writing messages)*

# Introduction

The article presents the main milestones of professional history of the authors, where the trans-institutional MY-X research team produced till now more qualitative ciphering solutions in order to demonstrate the quasi unlimited possibilities of transformations being capable of hiding information parallel to the general logic of private/public keys: e.g.,

* <https://miau.my-x.hu/miau2009/index.php3?x=e0&string=cipher>,
* <https://miau.my-x.hu/miau2009/index.php3?x=e0&string=aesthetic>,
* <https://miau.my-x.hu/miau2009/index.php3?x=e0&string=jav%C3%ADt%C3%B3kulcs>

The article also presents the necessary backgrounds/benchmarks for the own ideas: The well-known Caesar-codes define a small combinatorial space – easy to identify it. The Caesar-codes are always focusing on letters/numbers/characters. But the digitalisation represents these characters in form of pixel-matrices. Dot-matrix-fonts in dot-matrix-displays let flash certain pixels for a particular letter. Therefore, digital/binary Caesar-codes could be created if the active and passive pixels (see 1;0) would be modified – in a rule-based form.

The main part of this article demonstrate a few scenarios/examples, how the pixel-based ciphering can be constructed and what kind of limitation could be detected: The focusing on the pixels of the digitalized letters makes it possible to involve new transformations into the coding-decoding processes: e.g., pixels for one letter or even for arbitrary letters (see an entire picture – where in case of hand-writing-messages the recognition module can also be part of the decoding process) can be transformed into a circle in different ways (c.f. permutation), where the Caesar-code-like shifting can be interpreted immediately. Not only simple shifting-patterns can be defined, but also rules for cellular automata. These rules can even be so complex, that the decoding is quasi impossible, or this becomes a new combinatorial challenge. The digital representation of letters can also be seen as a door to the genetic algorithms with their specific transformations (like mutation, crossing, etc.). Patterns for binary pixel-values can be created based on different series like Fibonacci or even specific randomized inputs like pi. Digitalized pixel-values can have numeric contents (like in case of grey-scale-pictures). With these numeric values, new (quasi arbitrary) transformations can be initialized. The above-mentioned possibilities are not direct concurrent approaches for the well-known cryptographical solutions (see private/public keys), where the quantities become qualities (in case of limited computational capacities).

Finally, the article describes the potential perspectives of the pixel-based de/coding: The pixel-oriented approaches can rather be compared to the old/rare native languages used e.g., in the second world war e.g., by Americans e.g., against Japanese (c.f. code talkers). Therefore, the above-presented pixel-based approaches can be used in case of specific demands: communication between business partners, agents. These qualitative techniques make possible to hide graphical plans as a kind of non-figurative picture. Specific watermarks can also be created based on pixel-transformations. These solutions can be combined with quasi arbitrary other effects (c.f. letter-statistic-homogenisation, blurred contours of letters, more substitutions for one original input, etc.).

# Previous projects

The challenges of cryptography can be seen as a specific support for other problem-solving tasks:

1. <https://miau.my-x.hu/miau2009/index.php3?x=e0&string=cipher>,
2. <https://miau.my-x.hu/miau2009/index.php3?x=e0&string=aesthetic>,
3. <https://miau.my-x.hu/miau2009/index.php3?x=e0&string=jav%C3%ADt%C3%B3kulcs>

[1] The first project URL (\*cipher) presents 3 articles where the telephone-keyboard inspired a case study for education of Students on the field of the IT-security. The abstracts of the articles demonstrate the main characteristics of the challenge and its solution:

“*The history of the ciphering offers a lot of solutions and theoretically, quasi unlimited ideas should be existing for causing problems during the interpretation of messages. This article is one of series having three parts. The first part presents a cipher algorithm created by an author having just a little experience in programming. The part II. demonstrates the appropriate decipher algorithm where the basic idea was delivered through the old-style mobile phone’s keyboard offering more letters per button. The last (third) part shows the same functionality like in the first part – however this source code got realized by a professional author. Based on the comparing of the more and less trained thinking methodologies (of more and less optimized source codes), it is possible to demonstrate the development possibilities concerning abstraction/complexity of approximations. A rel. robust cipher ensures that the starting and/or ending position of words can not be identified. It is also a sign of higher complexity of ciphering, if one particular letter in the raw message will be substituted through more than one code. Besides, it is necessary to have meaningless/random parts in the ciphered messages. Parallel, it is a more complex level, when the same message can be described through different ciphers in an identical way. The presented demo is a C++ program which can be used through the online services of repl.it. The article also includes demo messages and their ciphered versions.*” (Source: <https://miau.my-x.hu/miau/254/cipher1.docx>)

“*Based on the principle of KNUTH, it is just possible to speak about deciphering, if the whole process could be transferred into source code and therefore it can be used in an automatically way. It means the intuitive solution (guessing) is rather a kind of artistic performance as knowledge – independent from its success. This article is the second part of a series having three parts altogether. The first part already presented the ciphering algorithm. Here and now, it is possible to see the deciphering algorithm. The third part will demonstrate the same functionality of ciphering but instead of a solution created by a less trained programmer, the last part will present a more professional (optimized/efficient) source code. One of the goals of the article is the support the gamified learning.*” (Source: <https://miau.my-x.hu/miau/254/cipher2.docx>)

*“During programming of ciphering algorithms, the authors will be confronted sooner or later with questions concerning code-efficiency. The ciphering/deciphering may not need unlimited time and/or computing resources in case of real processes where the communication itself also needs time and/or the disturbed communication can increase this demand in an extreme way. The efficiency of the source codes can be increased in different (partly parallel) ways. The third (final) part of the series a less and a more optimized source code will be compared where the functionality the same is.”* (Source: <https://miau.my-x.hu/miau/254/cipher3.docx>)

[2] The second challenge produced two articles about the AI-based handling of the context free beauty, where a side-effect led to the pixel-based de/coding as such:

*“The Vitruvian man of Leonardo Da Vinci is a kind of evidence of the specific ratio (divine proportion, golden ratio) being available in the nature (c.f. phi=1.1618). Parallel, the mathematics of the beauty has to derive a scale for robot-eyes. This scale should be capable of estimating/describing the difference between two pictures (here and now between a randomized pixel-set and a real nonfigurative painting) in a context-free way. The scale has to support the ranking of the beauty of pictures. It is therefore possible to derive beauty and its inverse, or even a good-better-best distinction between pictures. The kind of artificial intelligence-based beauty-definition has to lead to a positive Turing-test – without any training, without human evaluation of pictures in advance. The robot-aesthetic-expert (the robot-eye) is capable of sensing/estimating the ratio of the visual conception and the coincidence. This beauty-scale could be derived from the context-free risk-definition of the similarity analysis where risk-free is a data pattern where each data position can be derived from the rest. The same logic can be used for the definition of the harmony/beauty. A kind of side-effect is a visual risk-map. This map gives red alert signals if the interpretability of a part of the picture is less than the interpretability of the neighbourhood. The risk-map can be used by artists to improve the compositions – like sounds of a less trained singers can be optimized based on automated adjustments processes.”* (Source: <https://miau.my-x.hu/miau/270/roboteszteta.docx>)

*“The I. part about the robot-aesthete presented the theoretical backgrounds concerning the mathematics of the beauty as such. This part (II) demonstrates a Turing-test-process, where 15 random-selected, but nonfigurative pictures got evaluated by a random set of human beings (by 36 people) on an artificial beauty-scale (1<10) in a separated evaluation process for coloured and grey-scaled versions. The analysis of the human evaluation should deliver a statement for the question: Is it possible to create a scale for measuring the randomness of the human answers concerning each picture? This quasi classic (statistic-oriented) challenge should be solved based on similarity analyses – based on the same AI-engine what will be used for further analyses concerning the beauty as such. This randomness-scale will let derive: which picture could be evaluated in the most robust way (it means: less random-like) and which picture generated the most random-like human evaluations. Therefore, we will also know, which pictures are under-norm, over-norm or even norm-like concerning the randomness of the human evaluation. All these will be derived without using the logic of significant differences. Based on the raw human evaluation, a kind of aggregated (similarity-oriented) beauty-index will also be calculated. Parallel, a naïve robot-eye will also be constructed being just capable of estimating grey-scale-values for each pixel based on the neighbourhood (it means: on 3\*3 pixels). The estimated pixels and the real pixels lead to differences, where the more beautiful is a picture, the less is the volume of the differences (describe through a lot of statistical attributes). The results: 3 of the analysed pictures could be evaluated as valid objects because of random-like human evaluations and/or modelling anomalies. There is one single picture where the naïve robot-eye and the human beauty-conceptions lead to the same conclusion: this single picture is an under-norm-object concerning the pattern-oriented beauty-definition. Therefore, it is not a trivial successful Turing-test – but the success is more depending on the human beings as the naïve robot-solution.”* (Source: <https://miau.my-x.hu/miau/270/roboteszteta_II.docx>)

The side-effect of the seemingly totally different research objectives is simple: if a pixel-based message will be decomposed to “n” (identical dimensioned raw & column) layers with grey-scale-values then e.g., the sum of these values will lead to a (n+1). pixel-value. If the “n” entire non-figurative (or even partially interpretable/figurative) layers can be sent to the targeted groups, the these layers without any information or even with traps can never be decoded, but the sum of the n layers let make visible the original message as such.

[3] This project demonstrates the possibility of decoding based on Solver-engines. The source-URL let make visible a set of XLS-files which presents the Solver-based potentials for code “breaking”. The challenge, the apropos of this solution-family came from a group of Students having scored tests without the possibility to see the correct answers for each question. If each Student stored the own answer-set, then it is possible to derive the hidden code for the scoring based on a Solver-engine. (Source: <https://miau.my-x.hu/miau2009/index.php3?x=e0&string=jav%C3%ADt%C3%B3kulcs>)

# Backgrounds and benchmarks

Quasi each cryptography education starts with the Caesar-code and its cracking, where the sequential limited alphabet will be transformed into a circle in order to use an integer value for coding/decoding (it means for shifting).

Parallel, the pixel-based logic can be reformulated as a kind of “digital/binary coding” and these phrases can be identified in a lot of publications (like: <https://www.ms.sapientia.ro/~mgyongyi/Crypto/Labor1.html>, <https://www.hit.bme.hu/~buttyan/publications/bscinfkod.pdf>).

The association space about the term of binary coding makes it possible the binary characteristics of arbitrary objects to analyse. Concerning the letters/numbers, the binary version is a kind of dot-matrix. On the other hand, black & white pictures can also be interpreted based on binary basic information. If an interpretation space for binary values is given, it is mostly also possible to define integer values (see grey-scale-values of pixels between 0-255) for the same “variables” (pixel-positions).

The Caesar-code and the digitalisation of letters/numbers (characters) let change the focus of the de/coding-processes from a character to a pixel.

On the field of the potential/creative transformations of binary pixel-values, the next association could be the term of the genetic algorithms, where binary values needed to the representation of a solution can be changed (c.f. mutation, crossing – e.g., <https://miau.my-x.hu/miau/244/ga_excel.xlsx>).

And finally, the pixels in a picture and/or in a dot-matrix can also be interpreted as cells of a systems with 2D-relationships. This leads to the term of cellular automata (c.f. <http://miau.my-x.hu/miau/237/cellu1.xlsm>), where the binary pixels as cell-information can be change based on more or less simple rules.

Summa summarum: each component for a pixel-based ciphering-LEGO-game is now given. The next chapter presents the already closed experiments to explore the real possibilities and the real limitations:

# Scenarios/versions

1. Binary Caesar-code within a character-based dot-matrix with circular patterns: see <https://miau.my-x.hu/miau/303/crypt1.xlsx>
   1. Well-known printed fonts (sheet: letter-universe1 & letter-universe2): the circle-building has quasi unlimited alternatives with different risks and computational needs (see figure#1). Circles can be built based on quasi unlimited alternatives (raw-oriented, column-oriented, pattern-oriented like first the even ones and then odd ones or vice versa, or even division through a number being not exactly a precise divisor of the circle-length, etc. The coding based on the same transformation logic can be repeated (see figure#1 – e.g., bottom part) – the repetition value can be derived e.g., from PI or even a series (like Fibonacci, where always the last letter-position value should be accepted – it means, we need just 10 repetitions between 0-9.
   2. Hand-written characters: in case of hand-written characters, it is not even relevant having scaled/normalized fonts. Real hand-written texts (where the originality of the authors could parallel be checked) can be sliced into n\*m dot-matrices which can be transformed like in case of 1a (above). Outputs of the coded messages can have borders built from arbitrary pixels. Therefore, the above-mentioned slicing is a further layer of disturbing effects. Letters can have different dot-matrices too, where the types of a dot-matrix can be regulated through PI or other series (like Fibonacci – e.g., the last (two) character-positions are the raw- and column-coordinates).





Figure#1: Simple (top) and complex (bottom) circle-building patterns for one single letter-dot-matrix (source: cryprt1.xlsx)

1. Two or more digitalized characters (parts of a picture) as a common dot-matrix
   1. It is not more needed to present new figures (like #1) based on the scenarios 1a & 1b, in order to declare, that the dot-matrices can be arbitrary small or extensive. It is possible to code (transform) not only one single character, but arbitrary characters with one transformation in a parallel way, where the original object (dot-matrix) describes arbitrary letters/numbers (like hand-written option – c.f. Chinese signs). The parallel coded inputs will be decoded also in a parallel way.
   2. The block-wise (letter-wise) building of common dot-matrices is not the single way: a holistic picture can be sliced in an arbitrary way (c.f. Tetris-patterns, series-based patterns).
2. Cellular automata
   1. Decodable transformations: The binary Ceasar-code interpreted for dot-matrices as such can be interpreted as a rule for a cellular automaton. The rule is simple: the new value of a cell should be some of the values of the neighboured cell. It is relevant to highlight that the rule in case of one single neighboured cell can be decoded in a trivial way.
   2. Non-decodable transformations: In case of a rule with more than one involved (neighboured) cells, it is not more trivial, that the decomposition/decoding can be executed in an exact way. For example: the new value of a cell should be 1, if the sum of the values in the neighboured (e.g., 8) cells is even and the new value of the cell should be zero, if the sum (built like above-described) is an odd number. (The same logic is valid e.g., for modulo-oriented rules). The binary output values (1;0) come from more than 2 input cases (2^8), therefore it is not trivial to derive from an output value, which input combination stood behind it. On the other hand, it is possible to create decodable rules based on more than one input cells involved in a rule: if we avoid binary outputs (see sheet: cellular automaton3 – Figure#2). Non-decodable transformations are all transformations, where the input values have more than one output-substitutes (see sheets: cellular automaton1&2 – Figure#3). Quasi unlimited undecodable transformations can be defined e.g., the binary content of the focused (neighboured) cells can build different sequences from 0 and 1 like a classic binary value and these binary values can be transformed into 1 or 0 depending on their characteristics like hexa-code has numbers or not…



Figure#2: Decodable cellular automaton – rule = sum(column)+3\*own-value (source: crypt1.xlsx)





Figure#3: Antagonisms in the outputs (source: crypt1.xlsx) – Legend: rule for the top scenario = sum(column)+own-value, rule for the bottom scenario = sum(column)-own-value, example for antagonisms with borders

Remarks: The previous transformations can be involved after a decodable cellular automaton as transformation…

1. Genetic algorithm-like transformations
   1. Instead of mutations: each input cell-pattern should be inverted, where a cell-pattern could be a 2\*2 cell-object (see Figure#4 – sheet: mutation). Here and now, it should already be trivial, that all titles like mutation, cellular automaton, circle-building, etc. are totally irrelevant – there is only transformation rules (decodable or non-decodable).
   2. Instead of crossing: the 2\*2-pattern could also be titled as a crossing (within the 2\*2-pattern), but Figure#5 presents a partially (top) and entire (bottom) wide-ranged crossing patterns (see sheet: crossing):



Figure#4: 2\*2-pattern-mutations (source: crypt1.xlsx)



Figure#5: Partial/entire, wide-range crossing-actions (source: crypt1.xlsx) – Legend: top-pattern with one single 4\*4 transposed part, bottom-pattern with 4(4\*4) transposed patterns (source: crypt1.xlsx)

1. Decomposition of pixel-based characters
   1. Randomized decomposition: The figure#6 presents a decomposition example, where 3 randomized layers are integrated into the final picture. Randomized effects leads in very low frequency to useful messages (see 4th picture – central on the top part).
   2. Optimized decomposition: If the input (the so-called 4th picture is predefined, then the decomposition can also be executed with randomized values: the first value can be generated in the entire interpretation space, the second value in the interpretation space reduced through the first value, and the last value is a kind of residuum (see bottom part of figure#6 with a control picture). The interpretation space can be optimized for grey-scale-intervals (0-255). The decomposed layers can always be sent as a non-decodable message. The code-crackers will never know based one of the decomposed layers: how many layers are necessary to one picture and how to integrate the decomposed layers.

Ein Bild, das Quadrat, Kreuzworträtsel, Screenshot, Rechteck enthält.

Automatisch generierte Beschreibung

Figure#6: Randomized and optimized decompositions (source: crypt1.xlsx) – Legend: top = randomized, bottom = optimized versions

A secret-key-like approach is if one or more decomposed layers are already given by the receivers, and the senders know these layers and the senders can involve them into the decomposition processes. The number of the sent layers can be reduced, it means that the code-breaker will never see all the components of a messages.

# Discussion

The character-(letter/number)-based coding needs less resources as a pixel-based transformation approach. Therefore, the here and now presented solutions are worth using in such cases, where short messages and/or important pictures should be protected in a qualitative/rare/unique way.

The above-mentioned solution layers can mostly be combined with each other. Therefore, the combinatorial space can be increased quasi unlimited to cost of the computation time.

The here and now not (detailed) listed characteristics as weaknesses of potential coding techniques can also be handled (e.g., the letter-frequency can be equalized through additional original text parts without any meaning, or the original text can be designed with repetitive letters in adequate volumes where the repetitions do not disturb the interpretations (e.g., innterppretaatiooon).

The original letters/numbers can be represented not only through the same dot-matrices.

# Conclusions

The rarity of this kind of core ideas and the ab-ovo rare using of a technique, makes quasi impossible to crack such kind of codes.

The preparation phase between the affected persons/systems can not be reduced to zero time. Coding strategies have to share in advanced with each other.

# Future

The non-decodable transformations should have a subset, where the rules can not be inverted, but it seems to be possible to derive Solver-based solutions like in case e.g., of test-correction-schemes. In future, at least one demonstration version should be derived in future.

The presented solutions are for education processes useful independent from the cryptographical market value of the solutions, where the students (interested in IT security) want to learn creating sources codes in case of matrix-dependent problems.

# References

Pitlik, L. el al (2023): Demo materials for a pixel-based cryptography, MIAU-Journal, No.303, <https://miau.my-x.hu/miau/303/crypt1.xlsx>

Pitlik, L. et al (2022): Derivation of evaluation test-codes based on solver in different case, MIAU-Journal, No.282-283, <https://miau.my-x.hu/miau2009/index.php3?x=e0&string=jav%C3%ADt%C3%B3kulcs>

Pitlik, L. et al (2021): Robot-aesthetic-expert or the mathematics of the beauty - Part, MIAU-Journal, No.270, <https://miau.my-x.hu/miau2009/index.php3?x=e0&string=aesthetic>

Pitlik, L. et al (2019): Cipher - exercises for advanced users – Part I-II-III, MIAU-Journal, No.254, <https://miau.my-x.hu/miau2009/index.php3?x=e0&string=cipher>

Pitlik, L. et al (2018): Genetic algorithms for problem solving - reproducible demo, MIAU-Journal, No.244, <https://miau.my-x.hu/miau/244/ga_excel.xlsx>

Pitlik, L. et al (2018): Simple cellular automata based on Excel-makros, MIAU-Journal, No.237, <http://miau.my-x.hu/miau/237/cellu1.xlsm>

Márton, Gy. (2023): Cryptography, Sapientia University, <https://www.ms.sapientia.ro/~mgyongyi/Crypto/Labor1.html>

Buttyán, L. et al (2006): Kódolástechnika, BME, <https://www.hit.bme.hu/~buttyan/publications/bscinfkod.pdf>

# Annexes

Source: <https://miau.my-x.hu/miau/303/crypt2.xlsx>





Figure#7: Letter-ASCII-Binary-Shifting-ASCII-Letter-method (source: crypt2.xlsx)

The letter-based (LABSAL) approach can be finetuned through the methodology presented in Figure#7. Letters will be changed, but the transformation rules are more complex than in case of the old Caesar-code. The same letter in the plain/original text must not be the same after the shifted transformations, but it is also not impossible (see spaces above in Figure#7). The LABSAL-approach needs less resources, but it has the same risks as the letter2letter-codes…

\*\*\*

Transcription-based cryptography – an idea originating from a randomized designed transcription functionality in TEAMS, where the Hungarian speaker’s voice is transcripted through a kind of English robot-ear. The output text seems to be a totally new language created by AI automatically. The question is: whether these ouput-texts can be decoded based on hard rules and/or based on learning patterns?! Demo: <https://miau.my-x.hu/miau/300/transcription-based-cryptography.mp4>