Indian Journal of Science and Technology, *Vol 7(9), 1403–1413, September 2014*
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Image Steganography using Ken Ken Puzzle for Secure Data Hiding

G. Vidya*, R. Hema Preetha, G. S. Shilpa and V. Kalpana

Sastra University, India; vidgopalakrishnan@gmail.com, preetharajen@gmail.com, shilugsnair@gmail.com, kalpana@cse.sastra.edu

Abstract

In image steganography, a number of data hiding schemes have been proposed and data has been hidden using them. Among these, the usage of mind-teasers and puzzles like Sudoku has also been used as schemes to hide data. In this paper, the territory of puzzle usage as data hiding scheme in image steganography is extended. A novel data hiding scheme based on Ken Ken puzzle is proposed. Though the Ken Ken puzzle is similar to the existing Sudoku technique of data hiding, the inherent difficulty involved in solving a Ken Ken puzzle is leveraged in this method and the final results after embedding data are then analyzed in terms of the visual quality of the stego-image obtained. This is done by means of certain quality metrics available to test the resulting image quality. The proposed system has been found to enhance the graphical quality of the resulting stego-image when compared to the existing system.

Keywords: Data Hiding Scheme, Data Security, Image Steganography, Puzzle-Based Steganography

1. Introduction

Everybody requires privacy in this world. The chances for tampering with the data have increased exponentially. The prominent growth in technologies puts privacy of data under constant threat. In the present scenario, security for communication is aimed at hiding the data. How the data hiding is brought about is called the data hiding scheme. Good data hiding schemes hide the data in an undetectable manner in other electronic media or "covers" like text, image, audio, video, etc. When data is screened or hidden inside a cover, it is called Steganography and one that employs an image as cover is called image steganography[1]. In image steganography, the major factors to be considered while building a data hiding scheme are[3]:

- The amount of data that is being embedded (embedding capacity).
- Data that is embedded should be imperceptible by the human eyes i.e., the data hiding scheme should produce lesser distortion in the image after embedding the secret data.
- The graphical quality of the stego image should remain nearly same as that of the original image taken.
- Should entail difficulty in extracting the secret information by constructing a scheme such that the embedding and the extracting involve a number of steps to crack the secret data hidden within the image.

In the work proposed, a novel data hiding scheme using another kind of mathematical puzzle like Sudoku called the Ken-Ken puzzle is carried out. The previous work is done using Sudoku as a data hiding scheme and has achieved good visual quality. In the present work, the visual quality rendered has come out with better results.

2. Related Work: Puzzles used in Data Hiding

Spatial domain steganography involves manipulation of pixels of a cover image in order to hide the intended data[4]. Many techniques for pixel manipulation in steganography have been proposed before. Initially, the Least

*Author for correspondence

Significant Bit (LSB) method was used where the Least Significant Bits of the image pixels were changed[2]. From then onwards, a number of other steganographic schemes have been introduced and recently many steganographic techniques employ puzzles like Jigsaw[5], Maze[6,7] and games like Tetris[8] as data hiding schemes. Though the idea of puzzle usage in steganography was motivated through the above methods, the major motivation for the proposed system was the Sudoku technique[3,15]. All the above methods use the puzzle itself as the cover medium within which the data is directly embedded except the Sudoku technique where the Sudoku is used as a scheme for hiding secret data inside a cover image rather than as the cover itself.

2.1 Existing Technique: Sudoku-based Data Hiding Scheme

A Sudoku is a square matrix typically of the order of 9. It consists of grids with digits such that horizontally (each row), vertically (each column) and in each 3*3 "block-wise" the digits never get repeated[9].

The main motive of this method is to manipulate the pixel pairs of an image to hide the secret data using an existing Sudoku solution. The steps involved in bringing about the data hiding are explained in this section.

2.1.1 Reference Matrix Generation

STEP 1:

Sudoku solution (Figure 1) is taken and is considered as a 9*9 matrix. Initially the values in the Sudoku puzzle lie between 1 and 9. All the entries in the Sudoku matrix are subtracted by 1 and this matrix is called as "TILE" matrix. This matrix contains values between 0 and 8.

0	3	6	1	2	7	4	5	8
1	4	7	0	5	8	2	3	6
2	5	8	3	4	6	0	1	7
3	6	0	2	7	1	5	8	4
4	7	1	5	8	0	3	6	2
5	8	2	4	6	3	1	7	0
6	0	3	7	1	2	8	4	5
7	1	4	8	0	5	6	2	3
8	2	5	6	3	4	7	0	1

Figure 1. Sudoku solution.

STEP 2:

The tile matrix (Figure 2) is further replicated to 27*27 matrix called reference matrix N.

2.1.1. Data Embedding:

STEP 1:

Secret information to be hidden into the cover image is converted to a Base-9 format. Let $D = D_1 D_2 D_3 D_4 \ldots D_n$ denote the converted secret data. Here n is the no of converted secret digits and $S_m \in [0, 8]$, $1 \le m \le n$.

STEP 2:

Embedding takes place by taking the red, blue and green components from all pixels consecutively. Every component is an 8-bit binary number which represents values from 0–255.

STEP 3:

This is further converted to a value using the following formula:

R1 = R1%9, G1 = G1%9, B1 = B1%9, R2 = R2%9
B2 = B2%9, G2 = G2%9

and so on. The above operation will result in value between 0 and 8.

STEP 4:

To make sure that the values are located in the centre of the reference matrix (Figure 3), 9 is added to the resultant values. Then the resultant (R1, G1) or (B1, R2) or (G2, B2) values become the row (X) and column (Y) indices of the reference matrix N, thus forming the pair (g_i, g_{i+1}).

STEP 5:

9 elements are chosen horizontally (CE_H), vertically (CE_V) and box-wise (CE_B) from the value at (g_i, g_{i+1}). CE_H (Horizontal row), CE_V (Vertical column) and CE_B (3*3 block), are called the candidate elements (Figure 4).

1	4	7	2	3	8	5	6	9
2	5	8	1	6	9	3	4	7
3	6	9	4	5	7	1	2	8
4	7	1	3	8	2	6	9	5
5	8	2	6	9	1	4	7	3
6	9	3	5	7	4	2	8	1
7	1	4	8	2	3	9	5	6
8	2	5	9	1	6	7	3	4
9	3	6	7	4	5	8	1	2

Figure 2. Tile matrix.

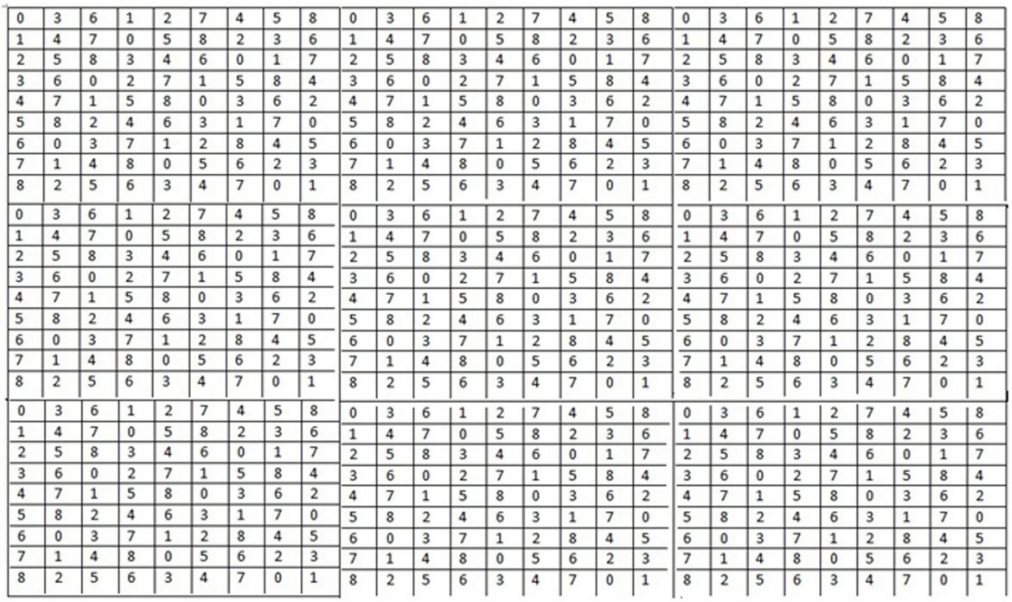

Figure 3. Reference matrix.

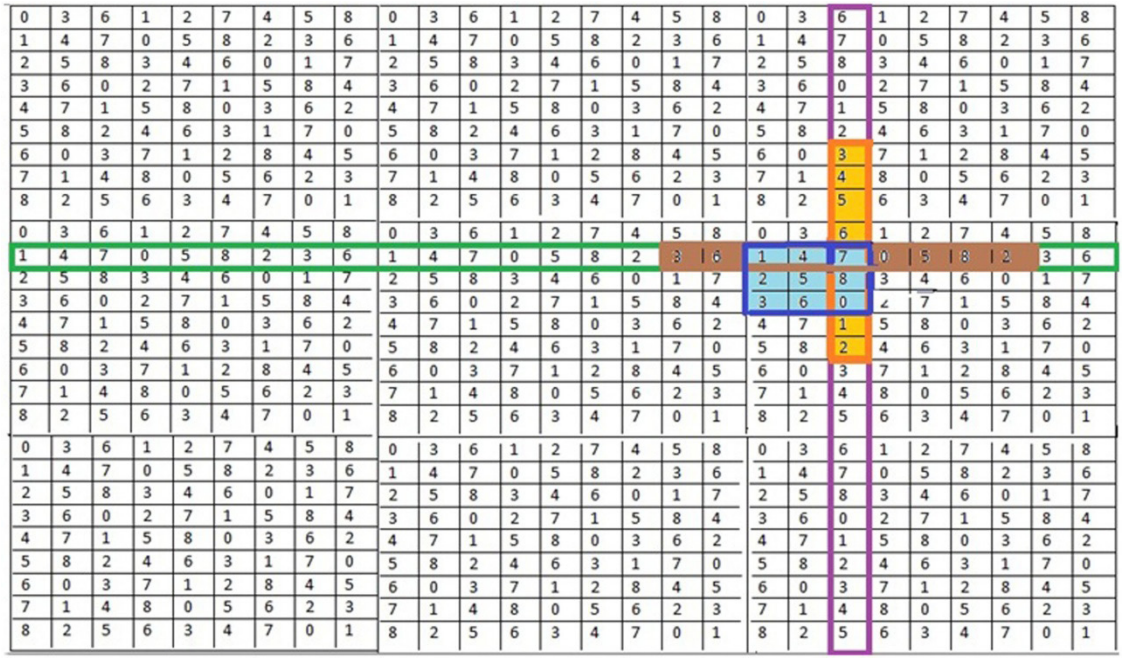

Figure 4. Replicated Sudoku matrix, brown(row), orange(column), blue(block of 3*3)

STEP 6:

Using manhattan distance formula:

$$N(x_{min}, y_{min}) = \min_{j} H, V, B \{|y_i - x_j| + |y_i + 1 - y_j|\}$$

a candidate element N (Xmin, Ymin) is selected with minimum manhattan distance from the value at (g_i, g_{i+1}). This candidate element is supposed to provide minimum distortion.

STEP 7:

Hence for concealing the secret digit D_i with less distortion, the pixel pair (g_i, g_{i+1}) is modified as (g_{i1} = Xmin, y_{i1+11} = Ymin).

2.1.2 Data Extraction

STEP 1:
The same Sudoku solution used for embedding can also be used for extraction.

STEP 2:
The pixels are taken and the reverse process takes place by taking (R1, G1) to extract the first digit, (B1, R2) second time, (G2, B2) the third time and so on. Their pixel values are converted to value between 0 and 8 using the formula

$$R1 = R1\%9 \quad G1 = G1\%9,$$
$$B1 = B1\%9 \quad R2 = R2\%9 \text{ and so on.}$$

STEP 3:
(R1, G1) or (B1, R2) or (G2, B2) values become the X and Y indices or row and column indices to be referred from the reference matrix formed from the Sudoku solution, thus forming the pair $(y_{i,} y_{i+1})$. The value at the position (y_i, y_{i+1}) is the secret digit.

STEP 4:
The same is repeated for all pixels until the entire secret data is extracted.

STEP 5:
The obtained secret digit is converted back to the normal format which gives the data embedded onto the cover image.

2.2 Proposed Technique: Ken Ken-Puzzle-based Data Hiding Scheme

A ken-ken puzzle can be built from any Sudoku if the concept of "cage" is being employed. It also employs the concept of grids with digits 1 to 9. The numbers inside the cages produces a 'target' number, that is obtained by performing some mathematical operations (addition, subtraction, multiplication, etc.,)[10]. The constraints for filling the numbers are:

1. The numbers present in a row should be unique.
2. The numbers present in a column should also be unique.
3. The numbers filled in a cage should fulfill the operation and the resultant number (this constraint is satisfied only if the operations or numbers are specified in the cage).

As a Ken Ken is loosely-based on Sudoku, it follows the same rules as that of a Sudoku and hence, every Sudoku is a Ken Ken with a standard cage size or "block-size" as said in Sudoku nomenclature. To bring out the difference however, between both these types of puzzles, in our scheme, we segregate the Sudoku puzzle created, into arbitrary shaped regions called "cages" and convert it to a Ken Ken puzzle.

The proposed algorithm follows similar modules as the existing method. The algorithm or procedure followed is explained under 3 modules.

I. Reference Matrix Generation of Ken Ken
II. Data Embedding
III. Data Extraction

2.2.1 Reference Matrix Generation of Ken Ken

Trial and Error method is used to easily generate a puzzle. Segregate the Sudoku into arbitrary shaped regions called cages and then later define operations and goals for those cages.

STEP 1:
Sudoku solution is generated and is considered as a 9*9 matrix. Initially the values in the Sudoku puzzle lie between 1 and 9.All the entries in the Sudoku matrix are subtracted by 1 and this matrix is called as "TILE" matrix. This matrix contains values between 0 and 8. Refer Figure 1 and Figure 2.

STEP 2:
The TILE matrix now formed from the Sudoku solution is divided into arbitrary shaped regions called "cages". This is done by one who intends to send the secret embedded data.

STEP 3:
The tile matrix with all the divided cages is further replicated to 27*27matrix called reference matrix M.

The operations and target numbers can then be defined separately by the user and included as a separate data file and sent to the intended extractor. This is done so that the intended extractor knows how to re-construct the Ken Ken solution in order to use the scheme to take out the data that was embedded into the picture or image. But the operations and target numbers do not play a role in the embedding process. So, the focus is mainly on the usage of the scheme built with cages and use of the same in actively embedding the intended secret data.

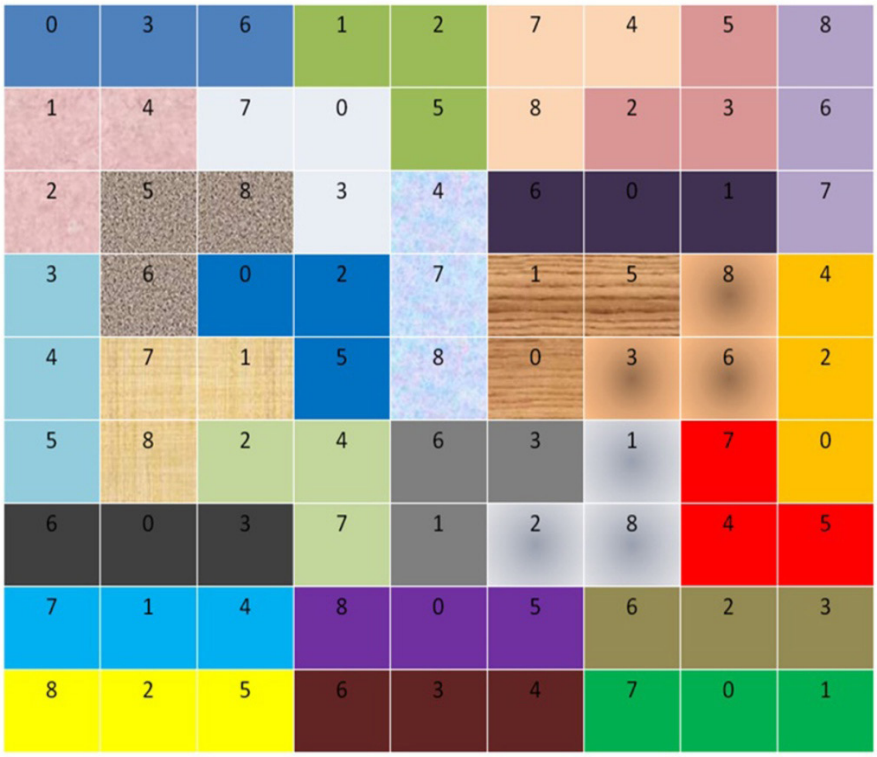

Figure 5. Ken Ken tile matrix.

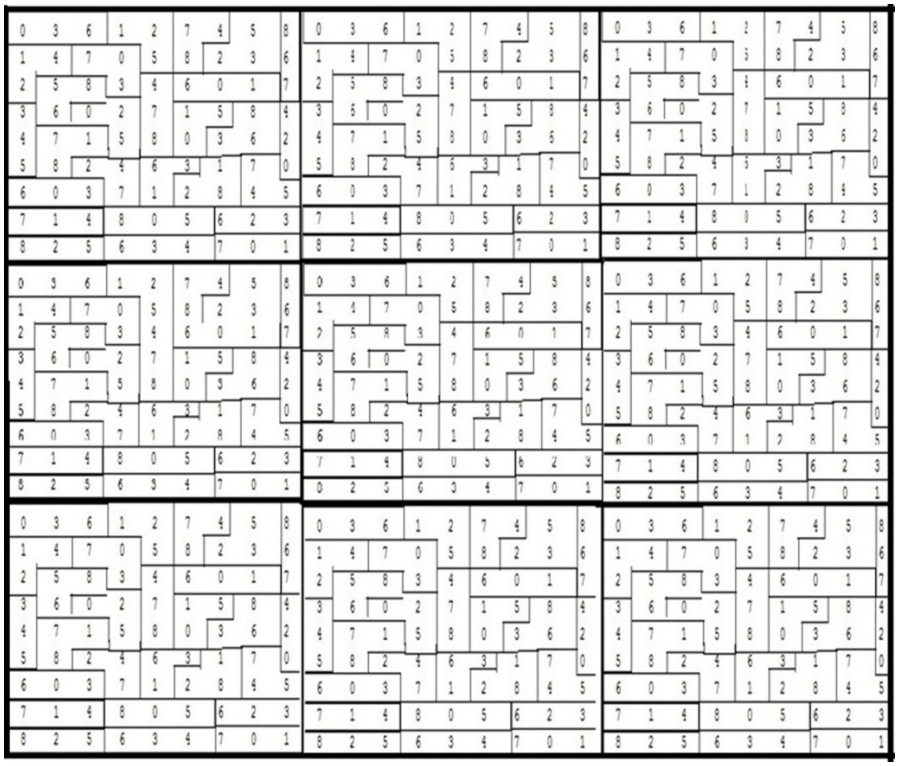

Figure 6. Reference matrix generated from the Ken Ken tile matrix.

2.2.2 Data Embedding

STEP 1:
Secret information to be hidden into the cover image is converted to a Base-9 format. Let $D = D_1 D_2 D_3 D_4 \ldots D_n$ denote the converted secret data. Here n is the no of converted secret digits and $S_m \in [0, 8]$, $1 \leq m \leq n$. [3]

STEP 2:
Embedding takes place by taking the red, blue and green components from all pixels consecutively. Every component is an 8-bit binary number which represents values from 0–255 as done previously[3].

STEP 3:
This is further converted to a value using the following formula:

R1 = R1%9, G1 = G1%9, B1 = B1%9, R2 = R2%9
B2 = B2%9, G2 = G2%9

and so on. The above operation will result in value between 0 and 8.

STEP 4:
To make sure that the values are located in the centre of the reference matrix, 9 is added to the resultant values. Then the resultant (R1, G1) or (B1, R2) or (G2, B2) values become the X (row) and Y (column) indices of the reference matrix M, thus forming the pair (g_i, g_{i+1}).

STEP 5:
The elements from the cage to which the central element belongs is checked first. In the best case, irrespective of whether we choose candidate elements from the row and column, we can directly take the position of the matched in the cage and modify the pixel value.

For eg., if 5 is the secret digit to be embedded, and the $(X, Y)^{th}$ element is the central element chosen here, in this Figure, 5 is present in the cage, the row and the column. But, since it is within the cage to which the central element belongs, its position is definitely the nearest in this case. Hence, the row and column need not even be checked for candidate elements like in the existing system.

Though this would work only in some cases where the digit to be embedded is always within the cage, we still take this into consideration. Also, when 9 elements from a 9*9 block is taken like in the existing system, we would not achieve as less distortion as we would when we choose a cage element which obviously is nearer and whose position when substituted in the original pixel value will produce lesser distortion. The results obtained

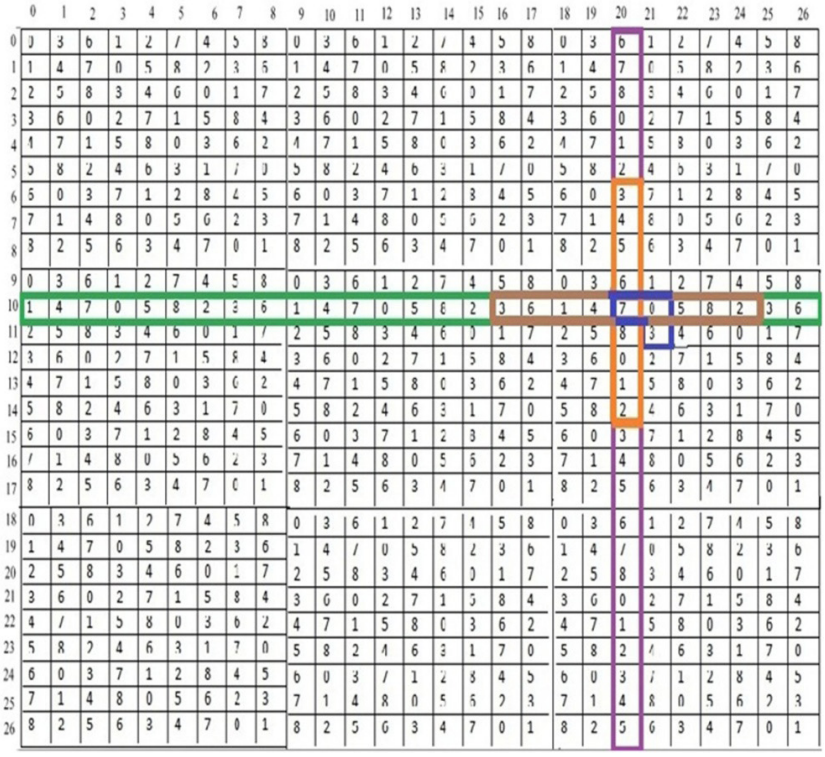

Figure 7. Replicated Ken Ken reference matrix, brown(row), orange(column), blue(central element's cage).

by implementing this have been recorded. These results prove to be nearly equal to the existing system (slightly better than the existing system in some cases) which validate thus, that the extended system also would work as well as the existing system (vide next section).

But if the element is not present in the cage, then we go in for checking the vertical column and horizontal row, consolidate the candidate elements as before, calculate the minimum manhattan distance and then finally choose the corresponding element.

STEP 6:

Using manhattan distance formula:

$$M(xmin, ymin) = minj = H, V, B \{| yi - xj | + | yi + 1 - yj|\}$$

A candidate element M (Xmin,Ymin) is selected with minimum manhattan distance from the value at (g_i, g_{i+1}). This candidate element is supposed to provide minimum distortion.

STEP 7:

Hence for concealing the secret digit D_i with less distortion, the pixel pair (g_i, g_{i+1}) is modified as $(g_{i1} = Xmin, y_{i1+11} = Ymin)$.

2.2.3 Data Extraction

STEP 1:

For extraction, the same Ken Ken scheme that was constructed to embed only has to be used. However, in order to add to the security, instead of sharing the solution fully, the person who embeds the data can just share the essentials for reconstructing the Ken Ken puzzle in a data file. The extractor who gets the shared puzzle manually solves the puzzle and re-constructs the Ken Ken using this data file. On obtaining the solution, they can generate the reference matrix as before and then start to extract the data.

STEP 2:

A data file containing strings is sent by the person who embeds to the person who extracts the data. This is used by the extractor to reconstruct the Ken Ken puzzle. Each string contains cage number, cage size, row (X) and column (Y) indices of the elements in the cage, target number and a number indicating the operation as shown in Figure 8 and 9.

STEP 2.1:

The specifications contained in the above text file is then used by the intended extractor to manually create the puzzle as shown below in Figure 10. This puzzle now needs

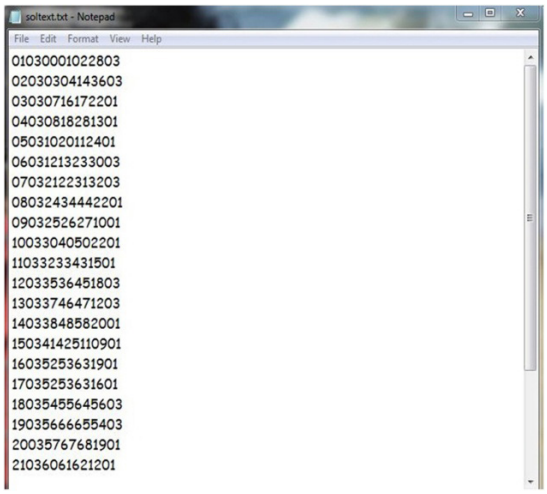

Figure 8. Data file to build Ken Ken puzzle.

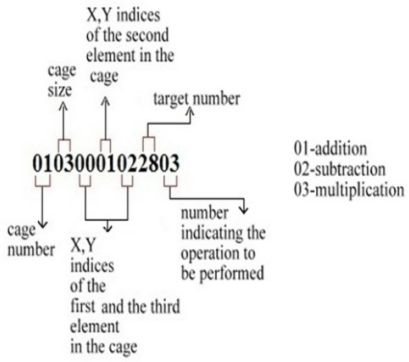

Figure 9. Structure of strings present in the data file.

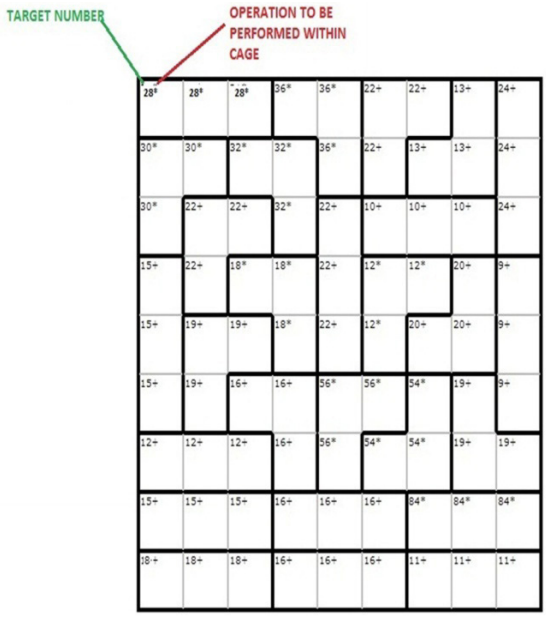

Figure 10. Unsolved Ken Ken built and shared to the extractor for obtaining solution.

to be solved by the extractor manually in order to obtain the correct scheme to extract the data that was embedded. In essence, after solving the puzzle, the extractor has to obtain the same scheme as that which was used by the person who embedded the data.

STEP 2.2:
The puzzle generated is solved manually by the intended extractor and the same Ken Ken solution that was used in embedding is obtained (Figure 11).

STEP 2.3:
Once the puzzle is solved, this is converted into a reference matrix as shown in Figures 5, 6, 7.

STEP 3:
The pixels are taken and the reverse process takes place by taking (R1, G1) to extract the first digit, (B1, R2) second time, (G2, B2) the third time and so on. Their pixel values are converted to value between 0 and 8 using the formula

R1 = R1%9 G1 = G1%9,
B1 = B1%9 R2 = R2%9 and so on.

STEP 4:
(R1, G1) or (B1, R2) or (G2, B2) values become the X and Y components of the Ken Ken reference matrix N, thus forming the pair (y_i, y_{i+1}). The value at the position (y_i, y_{i+1}) is the secret digit.

STEP 5:
The same is repeated for all pixels until the entire secret data is extracted.

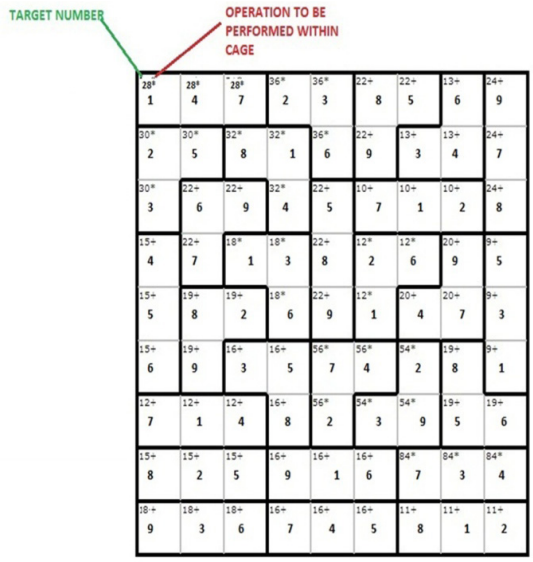

Figure 11. Solved Ken Ken.

STEP 6:
The obtained secret data is converted back to the normal format (Base-2 if bytes have been embedded or ASCII if text is embedded) which gives the actual data embedded onto the image.

3. Results and Discussion

The graphical quality of images obtained through this scheme was analyzed by means of certain image quality metrics and the values that the proposed system renders proves to be slightly better. This accounts for the proposed system to enhance the image quality to a certain extent. For analysis, a cover picture (image) with 1024 by 768 pixel resolution is taken. Some of the quality metrics considered to measure the resulting stego image quality are explained below:

Metric 1: Mean Square Error (MSE)

It is the collective squared error when the original and the stego images are compared. If the error has to be low, this value has to be low[14].

$$MSE = \frac{\sum_{P,Q} [I_1(p,q) - I_2(p,q)]^2}{P^*Q}$$

P and Q denote the number of pixels row-wise (horizontal) and column-wise (vertical) in the input images, respectively, I_1 and I_2 are the original and stego-images respectively.

Metric 2: Peak Signal to Noise Ratio (PSNR)

This is employed as a measurement metric to compare the graphical quality between the original and the stego images. As PSNR increases, the graphical quality of the image also increases[14].

$$PSNR = 10\log_{10}(\frac{R^2}{MSE})$$

It is considered that R is 255 since the input image is being taken accordingly.

Metric 3: Structural Similarity Index Metric (SSIM)

It works on the concept that the human graphical system is tailored highly to process structural information. The

algorithm for calculating this metric measures the change in this information between the original and the obtained stego image after embedding[11].SSIM is found to be a better quantifier of subjective quality than MSE and PSNR[12].

$$SSIM = \frac{(2^{*}\overline{I}_1{}^{*}\overline{I}_2 + C1)(2^{*}s_{I_1I_2} + C2)}{(s_{I_1}{}^2 + s_{I_2}{}^2 + C2)^{*}((\overline{I}_1)^2 + (\overline{I}_2)^2 + C1)}$$

where $\overline{I}_1$ and $\overline{I}_2$ are the mean of pixels in images I_1 and I_2 s_{I_1}, s_{I_2} are the variances calculated for all pixels in both the images and $s_{I_1I_2}$ stands for the co-variance between both I_1 and I_2 and C1 and C2 are constants.

Metric 4: Average Difference (AD)

It's the mean of the dissimilarity that results on comparing an original image with its stego version i.e., the image that carries the secret data. It is expressed by the following equation[13].

$$AD = \frac{1}{P^{*}Q}\sum_{i=1}^{P}\sum_{j=1}^{Q}(I_1(i,j) - I_2(i,j))$$

Metric 5: Maximum Difference (MD)

On comparing an original image with its stego version, it is the maximum possible difference that can result[13–14].

$$MD = MAX\left|I_1(i,j) - I_2(i,j)\right|$$

In the proposed system MSE, PSNR, SSIM, AD, MD values were taken and tabulated after embedding files of different sizes as shown in Table 1 (Figures 12–16).

The MSE, PSNR values that resulted suits the constraint "Higher the PSNR, lower the MSE"[14] (Table 2). Earlier works on steganography using Sudoku puzzle as a data hiding scheme has achieved good graphical quality of images[3]. In the proposed work, the graphical quality (Figure 17) rendered has come out with slightly better results. Even a minor increase of PSNR values, as evident from the graph, accounts for enhancement in the graphical quality of images. Also, in the Sudoku system, other statistical error measures such as Average Difference (AD), Maximum Difference (MD) and Human Visual System (HVS) based metrics like Structural Similarity Index metric (SSIM)[14] have not been determined. The proposed system has been measured in terms of those metrics too and the results have shown that the values obtained are above the threshold values. The threshold

Table 1. Various file sizes embedded and the recorded MSE, PSNR, SSIM, AD and MD values

Size of The File to be Embedded (in KB)	Metric 1 (MSE)	Metric 2 (PSNR)	Metric 3 (SSIM)	Metric 4 (AD)	Metric 5 (MD)
0.9208	0.0018	75.6795	0.9971	0.0864	255
1.05	0.0139	66.6952	0.9821	0.1026	255
5.34	0.0237	64.373	0.9542	0.1152	255
6.42	0.0365	63.5304	0.9232	0.1498	255
9.74	0.0915	63.112	0.8914	0.1962	255
31.2	0.311	54.035	0.8087	0.6987	255
42	0.342	53.721	0.7878	0.8688	255
52.2	0.361	52.5559	0.7571	0.8813	255
60	0.4267	52.0453	0.703	0.9002	255
74	0.4799	51.319	0.637	0.9081	255
86	0.5654	50.6074	0.5404	2.626	255
102	0.6745	49.8412	0.5161	6.6407	255

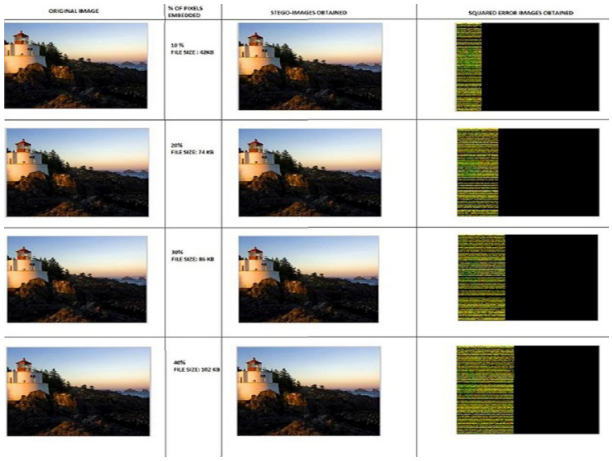

Figure 12. Stego images obtained after embedding sample files of varying sizes.

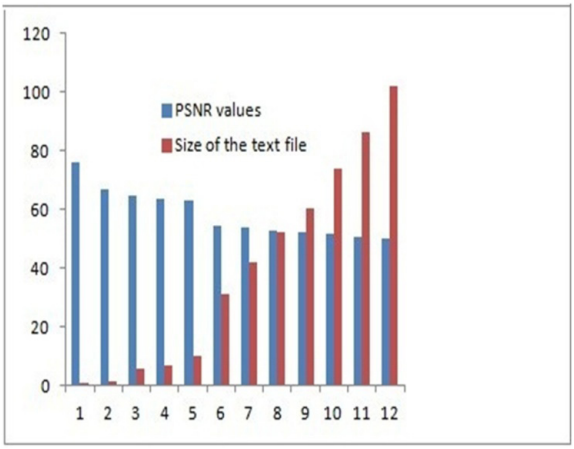

Figure 13. Graph of PSNR values as file size increases.

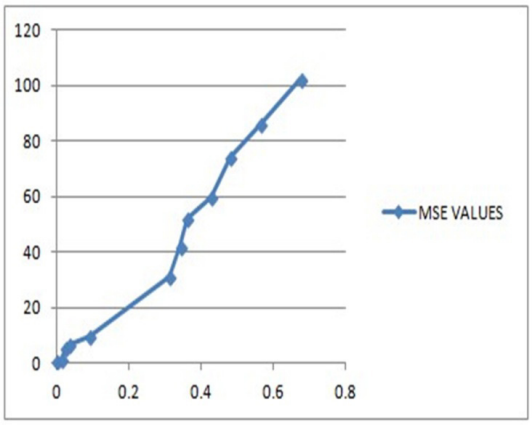

Figure 14. Graph of MSE values as file size increases.

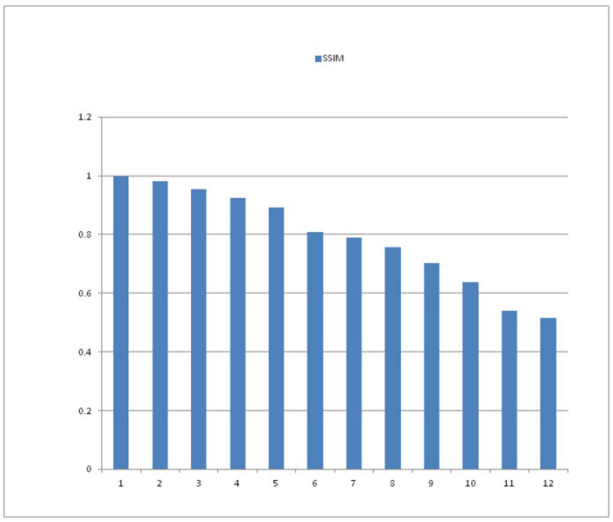

Figure 15. Graph of SSIM values as file size increases.

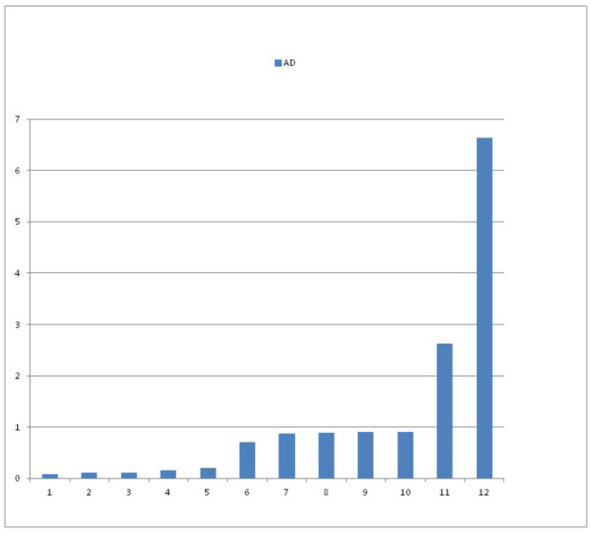

Figure 16. Graph of AD values as file size increases.

Table 2. Comparison between the existing system and the proposed system in terms of PSNR values

% of Pixels Embedded in Cover Image	PSNR Values Obtained in Existing Sudoku System[3]	PSNR Values Obtained in Proposed Ken Ken System
10	54.978	55.574
20	51.833	52.559
30	49.869	50.6074
40	48.836	49.8412
50	47.550	48.3128

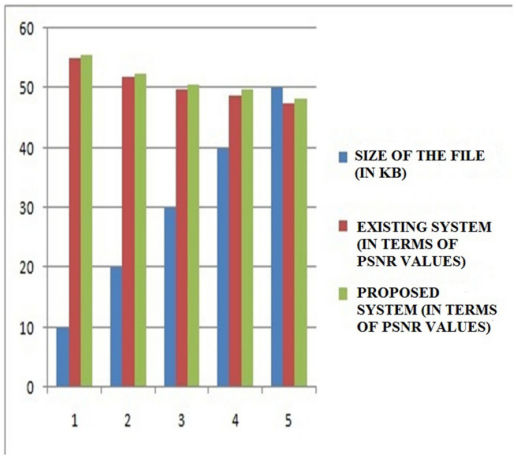

Figure 17. Graph between Sudoku and Ken Ken systems in terms of PSNR values

value is not distinctly defined for HVS based metrics like SSIM but by perception, the quality of the stego-image rendered is good.

4. Conclusion

On using the Ken Ken scheme for data hiding the following have been concluded:

i. Ken Ken scheme is built from a Sudoku. But, when it comes to extraction, the Ken Ken scheme has to be rebuilt by the intended extractor. This can be done only if they resort to solve the puzzle that was shared by the authorized sender, manually. The difficulty hence adds to the security.

ii. The image quality rendered by the Ken Ken scheme provides slightly better results than the existing Sudoku scheme. This is because on comparison, slightly higher Peak to Signal Noise Ratio(PSNR) values are rendered by the proposed Ken Ken scheme than the existing Sudoku scheme.

iii. Other statistical measures like Average Difference (AD), Maximum Difference (MD) are also calculated. These extra statistical quality measures calculated further prove the good graphical quality rendered by the Ken Ken data hiding system.

iv. Image quality is not only measured statistically but also based on human perceptive measures such as Structural Similarity Index Metric (SSIM). On checking SSIM too, the proposed scheme is found to be giving good results with respect to human perception of the resultant stego image.

v. The scheme thereby enhances the overall graphical quality of the image and also adds to the security because of the inherent d.ifficulty involved in solving the Ken Ken puzzle and then reconstructing the scheme.

5. Acknowledgement

The authors acknowledge and thank Dr. P. Swaminathan, Dean, School of Computing, SASTRA University, Dr. A. Umamakeswari, Associate Dean, Department of Computer Science and Engineering, SASTRA University and Dr. Manivannan D, Senior Assistant Professor, School of Computing (CSE), SASTRA University for their guidance and support.

6. References

1. Johnson NF, Jajodia S. Exploring steganography: seeing the unseen. Computer Journal of IEEE Computer Society. 1998 Feb; 31(2):26–7.
2. Morkel T, Eloff JHP, Olivier MS. An Overview of Image Steganography. Proceedings of the Fifth Annual Information Security South Africa Conference; 2005 June 29–Jul 1; Sandton, South Africa. Pretoria, South Africa: ISSA; 2005. p. 4–6.
3. Ijeri S, Pujeri S, Shrikant B, Usha BA. Image Steganography using Sudoku Puzzle for Secured Data Transmission. Int J Comput Appl. 2012 Jun; 48(17):31–5.
4. Zahidahadi S. Spatial Domain, Frequency Domain Time Domain and Temporal Domain [Internet]. 2012 Apr 10. Available from: http://ippr-practical.blogspot.in/2012/04/spatial-domain-frequency-domain-time.html.
5. Farn EJ, Chen CC. Jigsaw puzzle images for steganography. Opt Eng. 2009 Jul; 48(7):077006-1.
6. Lee HL, Lee CF, Chen LH. A perfect maze based steganographic method. J Syst Software. 2010; 83(12):2528–30.
7. Sukumar T, Santha. KR. Maze based data hiding using back tracker algorithm. IJERA. 2012 Jul; 2(4):499–504.
8. Ou ZH, Chen LH. Hiding data in tetris. Proceedings of the International Conference on Machine Learning and Cybernetics (ICMLC'2011); 2011 Jul 10–13; Guilin: IEEE; 2011. p. 61–3.
9. Delahaye JP. The Science behind Sudoku. Scientific American Magazine. 2006 June; 294(6):80–3.
10. Gerlach RJ. Solving KenKen puzzles—by not playing. Pharmaceut Program. 2010; 3(2):92–8.
11. Distler T. Algorithms for Image Quality Assessment (IQA) [Internet]. 2011 [cited 2013 Dec]. Available from: http://tdistler.com/iqa/algorithms.html.
12. Zhou W, Bovik AC, Sheikh HR, Simoncelli EP. Image Quality Assessment: From Error Visibility to Structural Similarity. IEEE Trans Image Process. 2004 Apr; 13(4):600–8.
13. Kumar R, Rattan M. Analysis of Various Quality Metrics for Medical Image Processing. Int J Adv Res Comput Sci Software Eng. 2012 Nov; 2(11):137–9.
14. Sasivarnan C, Jagan A, Kaur J, Jyoti D, Rao DS. Image Quality Assessment Techniques in Spatial Domain. IJCST. 2011 Sep; 2(3):177–81.
15. Shetty RBR, Rohith J, Mukund V, Honwade R, Rangaswamy S. Steganography using Sudoku Puzzle. Proceedings of the International Conference on Advances in Recent Technologies in Communication and Computing; 2009 Oct 27–28; Kottayam, Kerala, India. IEEE Computer Society; 2009. p. 623–6.