Revue Internationale du Chercheur
ISSN: 2726-5889
Volume 4 : Numéro 1

Optimisation de la capacité de stockage d'un QR code standard et sécurité de données

Optimization of the storage capacity of a standard QR code and data security

BOPE SAIDI Médard-Arnauld
Enseignant chercheur
Université Protestante de Lubumbashi
République Démocratique du Congo
arnauldbope@gmail.com

Date de soumission : 19/08/2022

Date d'acceptation : 01/02/2023

Pour citer cet article :

BOPE. S.A. (2023) « Optimisation de la capacité de stockage d'un QR code standard et sécurité de données », Revue Internationale du chercheur « Volume 4 : Numéro 1» pp :310 - 332

Résumé

Le QR Code est une technologie de stockage d'information dans une matrice 2D. (Denso ADC, QR Code Essentiels, 2011).
De nos jours, le QR Code est réputé comme pouvant encoder plus des données et de tout type contrairement au bar-code. Malheureusement sa taille aussi est limité (moins de 3KB) et ses données sont soumises aux contraintes de sécurité lorsqu'elles sont très sensibles.
C'est ainsi que quelques chercheurs proposèrent des solutions d'optimisation ou d'augmentation de la capacité du QR Code et des mécanismes de sécurité des données encodées dans ce dernier.
Cet article détail une analyse et conception d'un QR Code pouvant stocker jusqu'à quatre fois les données maximales qu'un QR Code standard peut encoder, plus important encore, de garantir la sécurité des données par l'utilisation des algorithmes de cryptage et des techniques de dissimilation des données.
Etant connue de tous et ouverte à tout le monde, il est aujourd'hui difficile de garantir la sécurité des données sensibles qui y sont encodées car sa création et décodage sont simple et accessible à tout le monde. Il devient alors nécessaire d'étudier et de proposer un QR Code sécurisé que nous qualifions de QR Code#.

Mots clés :

Limite QR code ; Optimiser QR code ; Sécurité des données ; QR Code# ; Stockage QR code

Abstract

The QR Code is a technology for storing information in a 2D matrix.
Nowadays, the QR Code is known to be able to encode more data and of any kind contrary to the bar-code. Unfortunately, its size is also limited (less than 3KB) and its data subject to security constraints when they are very sensitive.
This is why some researchers proposed solutions to optimize or increase the capacity of the QR Code and security mechanisms for the data encoded in it.
This article details an analysis and design of a QR Code that can store up to four times the maximum data that a standard QR Code can encode, more importantly, to ensure the security of the data by using encryption algorithms and data hiding techniques.
Being known by all and open to everyone, it is difficult today to guarantee the security of the sensitive data encoded in it because its creation and decoding are simple and accessible to everyone. It becomes then necessary to study and propose a secure QR Code that we call QR Code#.

Keywords :

QR code limit; Optimize QR code ; Data security ; QR Code# ; QR code Storage.

Introduction

Le QR Code (Quick Response Code) en français code à réponse rapide, est une technologie internationalement adoptée pour encoder des informations de tous types (caractères numériques et alphanumériques, des symboles, des caractères Japonais, des codes binaires et des codes de contrôle). Aujourd'hui, les informations codées dans un QR Code peuvent être stockées et transmises sous forme de texte simple, d'une URL (Uniform Ressource Locator), d'un numéro de téléphone, d'un SMS (Short Message Service), clé WIFI, etc. Et un utilisateur peut, avec un téléphone muni d'une caméra et d'une application de lecture du QR Code, afficher son texte, se connecter au réseau WIFI, lancer une page web dans le navigateur du téléphone, etc. (Skokov, Sneps-Sneppe, & Namiot)

Etant donné que le QR Code peut être généré et décodé facilement au moyen d'application informatique (sur téléphone smartphone ou sur le web), des personnes malveillantes (pirates informatiques ou hackers) peuvent y insérer des informations nuisibles (Peng, Sanabria, Wu, & Zhu). Ainsi, ces informations pourraient présenter des risques de sécurité soit pour les informations sensibles et privées encodé dans le QR Code (ex : code bancaire pour payement en ligne), soit pour l'appareil qui lit le code.

A ce problème de sécurité s'ajoute celui de la capacité de stockage du QR Code. Selon les spécifications du QR Code, la maximale que peut stocker un QR Code est d'environs 3KB (Denso ADC, QR Code Essentiels, 2011), ce qui lui permet seulement d'encoder de texte insignifiant en taille. Et si on voulait y encodes plus d'informations provenant par exemple dans une base des données, encoder une image (Rawat, Sahu, & Puthran, 2015), etc. cette taille devient vraiment insuffisante.

Etant donné que les technologies développées de nos jours visent de plus en plus à maximiser la capacité de stockage et à minimiser les risques de sécurité liés au traitement et acquisition des informations, nous nous posons une question de savoir comment la technologie sur le QR Code peut-elle rester compétitive en terme de capacité de stockage et de sécurité des données y encodées ?

Une transformation des données d'entrée (compression, cryptage et dissimilation) (Goel & Singh, Sep 2014) (Prathibha & Naresh, July 2014) (Rawat, Sahu, & Puthran, 2015) serait un moyen efficace de garantir un gain d'espace en même temps que la sécurité des données.

Nous proposons un algorithme permettant de multiplier les données d'entrée des QR Codes (par quatre) et cacher ces informations dans une image qui sera à son tour compresser et crypter

afin de contenir dans un QR Code. Cet algorithme sera ensuite implémenter pour produire une plateforme de génération et lecture de ce QR Code que nous qualifions de QR Code#.

Donc en générale, le QR Code# généré sera lisible par d'autres lecteurs qu'on trouve gratuitement mais malheureusement ne pourront pas en décoder les vraies informations que seul le lecteur approprier pourra fournir.

De ce fait, cet article vise à répondre à cette problématique en s'attelant dans la première section sur la revue de la littérature de QR code. Et nous présenterons en deuxième section le processus de génération de QR Code de Dipesh Rawat et Al que nous utilisons comme modèle de référence. A la troisième section nous allons étaler notre processus de génération du QR Code# qui constitue même la réponse à notre problématique. Une fois le QR Code# généré, nous allons présenter dans la quatrième section le processus de décodage et en fin à la cinquième section nous allons implémenter cet algorithme en java.

1. Etat de l'art sur le QR Code

Le QR Code à vue le jour au Japon en 1994 par DENSO WAVE suite à un besoin de stocker plus d'informations et de tous types (caractères Kanji, Kana et alphanumériques) que ne le permettait le code à barres (20 caractères alphanumériques) mais aussi pour gagner en termes de temps et réduire les erreurs de saisie manuelle. Pour sa première mise en application, le QR Code a été adopté par l'industrie automobile pour le suivi des pièces automobiles dans les processus de fabrications allant de la production, livraison, jusqu'à la publication des relevés de transactions (Denso Wave). Dès lors, le QR Code est devenu l'un de support de stockage le plus utilisé au monde surtout avec l'utilisation accrut de téléphone mobile mini de caméra (Rikala & Kankaanranta)

Le QR Code présente plusieurs avantages et inconvénients du fait de son coût de création et de lecture qui est quasi inexistant (Petrova, Romanello, B. Dawn, & Sandra, 2016), ce qui élargi son utilisation dans plusieurs secteurs comme : le marketing, le transport, la logistique, le système de santé, etc. (Denso ADC, QR Code Essentiels, 2011).

1.1. Caractéristiques

Un QR Code présente des nombreuses caractéristiques exposant son intérêt par rapport aux codes à barres traditionnel. (Denso ADC, QR Code Essentiels, 2011)

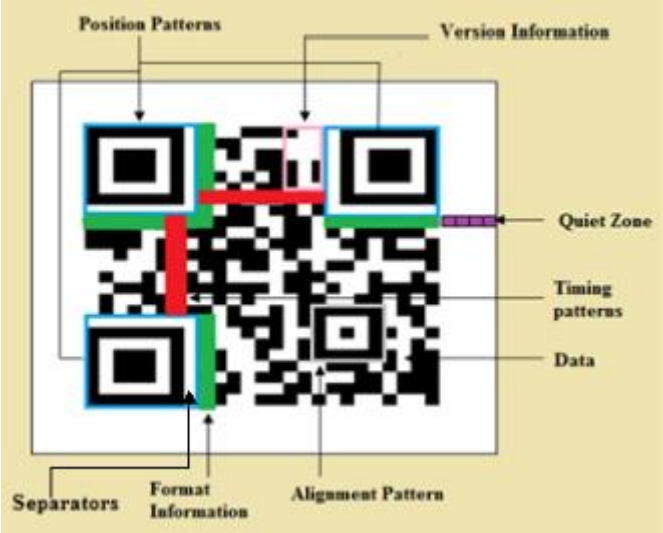
- Une capacité de stockage élevé : jusqu'à 7089 caractères,
- Une petite taille d'impression : pour la même quantité de données, le QR Code peut occuper environ le un dixième de l'espace du code à barres traditionnel,
- Une capacité à coder les caractères japonais (caractères Kana et Kanji),

- Une résistance à la saleté, à l'endommagement ou à la déformation : grâce à son code de correction d'erreur allant jusqu'à 30%,
- Une lecture facile dans tous les sens (soit 360°) : grâce à ses repères facilement détectables.

1.2. Structure

Le QR Code est constitué des plusieurs carrés noirs (1) et blancs (0) appelées modules.

Figure N°1. Structure du QR Code



Source : (Singh & Singh, June 2016)

- **Les motifs de détection de position (Position Patterns) :** Ils ont comme rôle de définir clairement l'emplacement et l'orientation du symbole dans l'espace. Ils sont au nombre de trois gros carrés situés dans trois coins du QR Code.
- **Séparateurs (Separators) :** ce sont des modules blancs autour des patterns de positions pour faciliter la distinction de ces patterns des données.
- **La zone silencieuse (Quiet Zone) :** zone non obligatoire, délimitant le QR Code et facilitant sa lecture. Elle est constituée de 4 modules en largeur.
- **Le pattern d'alignement :** permet de corriger la distorsion du QR Code lors de la capture de l'image.
- **Les patterns de synchronisation (Timing patterns) :** constitué d'une alternance de modules noirs et blancs et permet de fournir des positions de référence pour le calcul des coordonnées de chaque module.
- **Le module sombre** : un module noir unique qui est toujours placé à côté du modèle de recherche en bas à gauche.

- **La région de données (Data) :** c'est la partie du code qui contient les données ainsi que le code de correction d'erreur de Reed-Solomon.
- **Les informations sur la version :** renseigne sur la version ou la taille du QR Code. Ces versions vont jusqu'à 40 allant de 21x21 à 177x177 modules. La version la plus utilisée et la version 2 (25x25 modules).

1.3. Revue de la littérature

Le QR Code est réputé comme pouvant encoder plus des données et de tout genre contrairement au bar-code (Denso ADC, QR Code Essentiels, 2011). Malheureusement sa taille aussi est limité (moins de 3KB) et ses données soumises aux contraintes de sécurité lorsqu'elles sont très sensibles.

C'est ainsi que quelques chercheurs proposèrent des solutions d'optimisation de la capacité de stockage du QR Code et des mécanismes de sécurité des données encodées dans ce dernier.

Prathibha N. Pillai et K. Naresh (Prathibha & Naresh, July 2014) proposent un algorithme permettant de multiplier (jusqu'à trois fois) la taille des QR Codes en utilisant la technique de multiplexage et de codage par couleur. Ici, l'objectif est de prendre trois QR Codes, d'assigner aux modules noirs et blancs les valeurs du Rouge, Bleu et Vert pour enfin les combiner dans un seul QR Code en couleur.

Nancy Victor (Victor, 12-2012) et Sharu Goel et Ajay Kumar Singh (Goel & Singh, Sep 2014) proposent une technique de compression de données en entrée du processus de génération du QR Code pour augmenter la taille des données à encoder.

Dipesh Rawat et al (Rawat, Sahu, & Puthran, 2015) proposent un algorithme pouvant permettre au QR Code de stocker des images à haute résolution en implémentant un système basé sur la compression d'image et le cryptage. Les pixels de l'image sont convertis en une chaine des caractères compressés puis cette chaine est cryptée et encodé dans une séquence des QR Codes. Pour revenir à l'image originale, la séquence des QR Codes est décodée, décryptée et décompressée pour retrouver les pixels permettant de reconstruire l'image originale.

Kevin Peng et al. (Peng, Sanabria, Wu, & Zhu) Examinent les défauts de sécurité de QR Codes qui pourraient constituer un danger pour les utilisateurs et proposent des protocoles de sécurité pour éviter ces dangers. Ces protocoles consistent à encoder des bits cryptés ou signé numériquement via de des protocoles de sécurité traditionnels. Enfin, ils appliquent ces nouveaux protocoles sur les scanneurs Zxing (Bibliothèque open source du QR Code) et SpayD

(Bibliothèque open source de paiements par code QR) et comparent les résultats sur base d'une expérience sociale dont le phishing d'une URL.

Sankara Narayanan A. (Narayanan, 2012) présente différents types de données que peut contenir un QR Code, les attaques possibles sur ces données et proposent des solutions de sécurité à appliquer pour contrer ces attaques.

Julian Brackins et Mengyu Qiao (Brackins & Qiao), ont développés un prototype de sécurité qui authentifie les QR Codes en utilisant des fonctionnalités de sécurité intégrées. Leur système implique la modification du générateur du QR Code pour y intégrer le code d'authentification des messages et la lecture de ce QR Code sécurisé renverra normalement le message codé. Ensuite ils ont prévus une couche de validation du message en utilisation de systèmes cryptographiques.

Maykin Warasart et Pramote Kuacharoen (Warasart & Kuacharoen, 2012), présentent une méthode d'authentification rapide et convenable des documents basés sur des modèles (document administratifs, gouvernementaux, …) afin de permettre une vérification du document sans dépendre d'une quelconque institution de vérification. Cette méthode vise à compresser le document sur lequel un QR Code est imprimé et la signature digitale de ce document, puis encoder le tout dans le QR Code. Le décodage de ce QR Code consistera à décompresser le message, décrypté la signature digitale et comparer le message contenu dans la signature décrypté avec le message du document original.

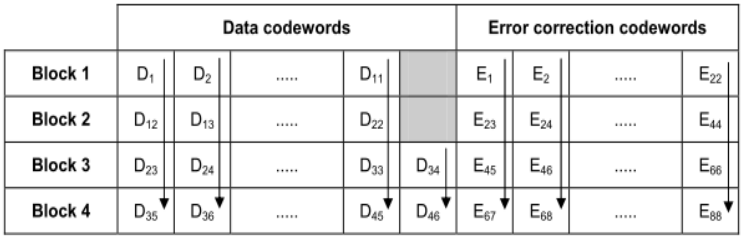
Jianfeng Lu et al. (Lu, et al., 2017) Proposent un système d'authentification des QR Codes utilisés dans un système de payement par téléphone en utilisant un mécanisme basé sur la cryptographie visuel et le QR Code personnalisé. Dans leur approche, ils scindent le QR Code original en deux ombres, qui par la suite sont assemblés chacun dans une image de fond et le résultat de chaque assemblage est fusionné avec un QR Code porteur. L'un de ce deux QR Codes est stocké sur un serveur en ligne et l'autre imprimé sur un support. Lors de la lecture du QR Code porteur, les deux parties du QR Code original sont empilés avec précision suite à un système de cryptographie visuel et enfin on obtient le QR Code original.

2. Processus de génération de QR Code proposé par Dipesh Rawat et al

Ayant étudié et compris le travail de Dipesh Rawat et Al (Rawat, Sahu, & Puthran, 2015) qui consiste à encoder une image cryptée dans un QR Code pour en optimiser la capacité de stockage et bien entendu la sécurité des données, cet article vient pour enrichir le leur en leur fournissant des données d'entrées, une image de moins de 100x100 pixels et dans laquelle les données sont quadruplées et cachées pour encore plus de sécurité.

En effet, la technique de Dipesh Rawat et al (Rawat, Sahu, & Puthran, 2015) permet d'utiliser d'une manière efficiente la capacité du QR Code étant donné que la capacité d'une image, surtout une image en couleur à haute résolution, dépasse de loin la taille d'un QR Code qui est de l'ordre de 3KB. Pour ce faire, ils proposent d'extraire les pixels d'une image en couleur, les convertir en un flux des données compressées puis cryptées pour produire une séquence des QR Codes comme l'illustre la figure ci-dessous :

Figure N°2 Construction de la séquence des mots de code du message final pour QR Code



Source : (Rawat, Sahu, & Puthran, 2015)

La compression et le cryptage sont assurés par le codage Base64 (Josefsson, Octobre 2006) qui consiste à coder toutes données codées sur 8 bits en utilisant des caractères imprimables au format US-ASCII (caractères non-accentués). Ce codage utilise un alphabet de 64 caractères (tableau 1 imprimables classiques représentant une donnée coder sur 6 bits et choisis pour être universellement lisibles et ne possédant pas des significations dans les principaux protocoles de messagerie.

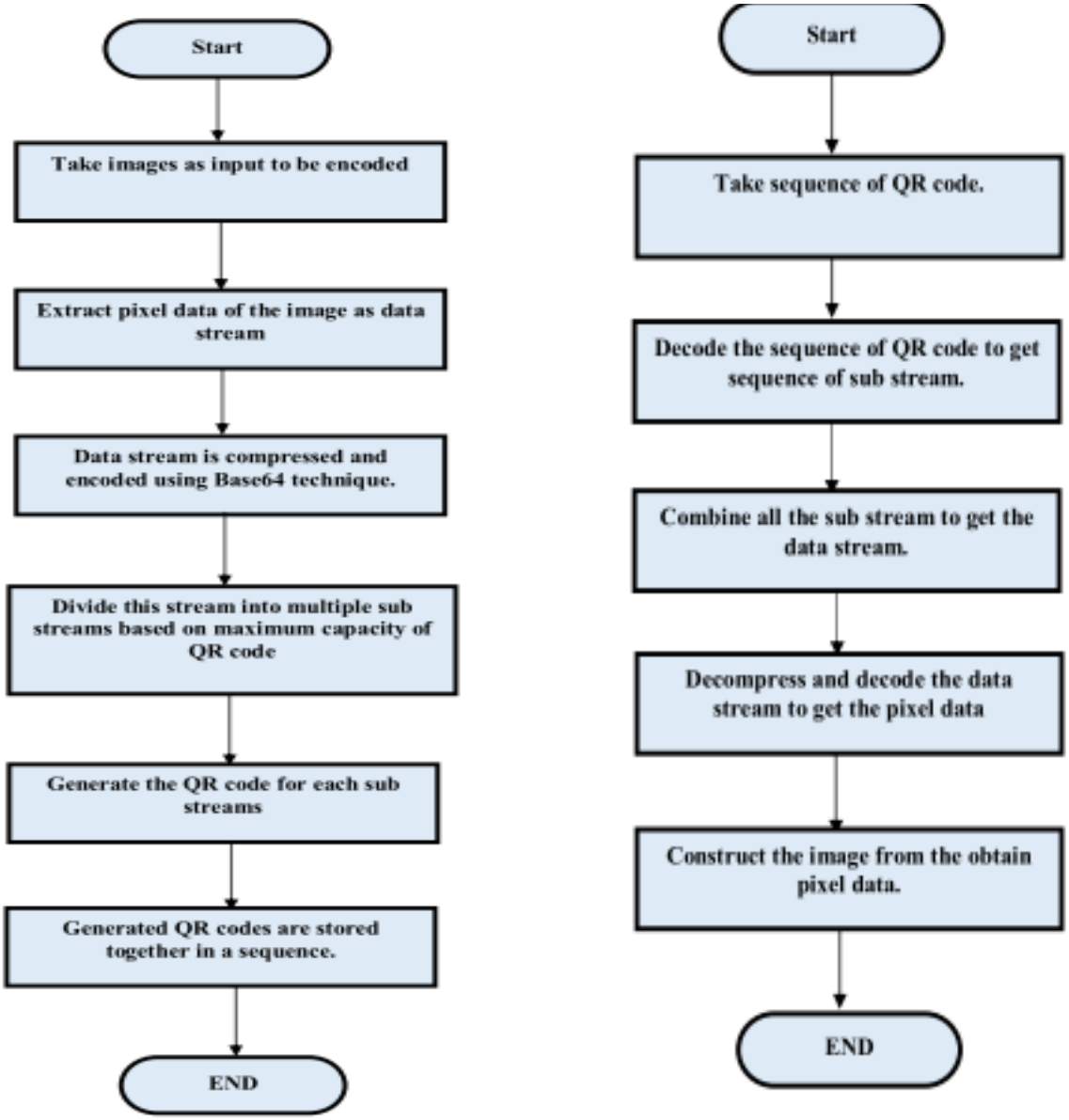
Tableau N°1 Caractères du codage Base64

Valeur	Caractère
0 à 25	ABCDEFGHIJKLMNOPQRSTUVWXYZ
26 à 51	abcdefghijklmnopqrstuvwxyz
52 à 61	0123456789
62 et 63	+/

Source : (Josefsson, Octobre 2006)

Les deux figures qui suivent, montrent deux images qui expliquent le processus d'encodage (Figure de gauche) et décodage (Figure de droite) du QR Code généré par la technique de Dipesh Rawat et Al.

Figure N°3 Conversion d'image en QR Codes et vise versa



Source : (Rawat, Sahu, & Puthran, 2015).

Le résultat obtenu est une séquence des QR Codes pour une image dont la résolution est supérieure à 100x100 pixels et un seul QR Code pour une image de résolution inférieure ou égale 100x100 pixels, comme indiqué dans le tableau ci-après.

Table N°2 : Résultat expérimental de Dipesh Rawat et al

Résolution de l'image	Nombre des QR Codes généré
100x100	**1**
400x400	**20**
1024x768	**82**
1440x900	**131**
1600x1200	**185**

Source : (Rawat, Sahu, & Puthran, 2015)

Eu égard aux résultats représentés dans le tableau N°2, notre étude s'intéresse au premier résultat qui produit un seul QR Code pour une image de résolution inférieure ou égale à 100x100 pixel. Dans la suite, nous allons présenter notre technique produisant cette image pour enfin l'appliquer dans la technique de Dipesh Rawat et al (Rawat, Sahu, & Puthran, 2015).

3. Processus de création du QR Code#

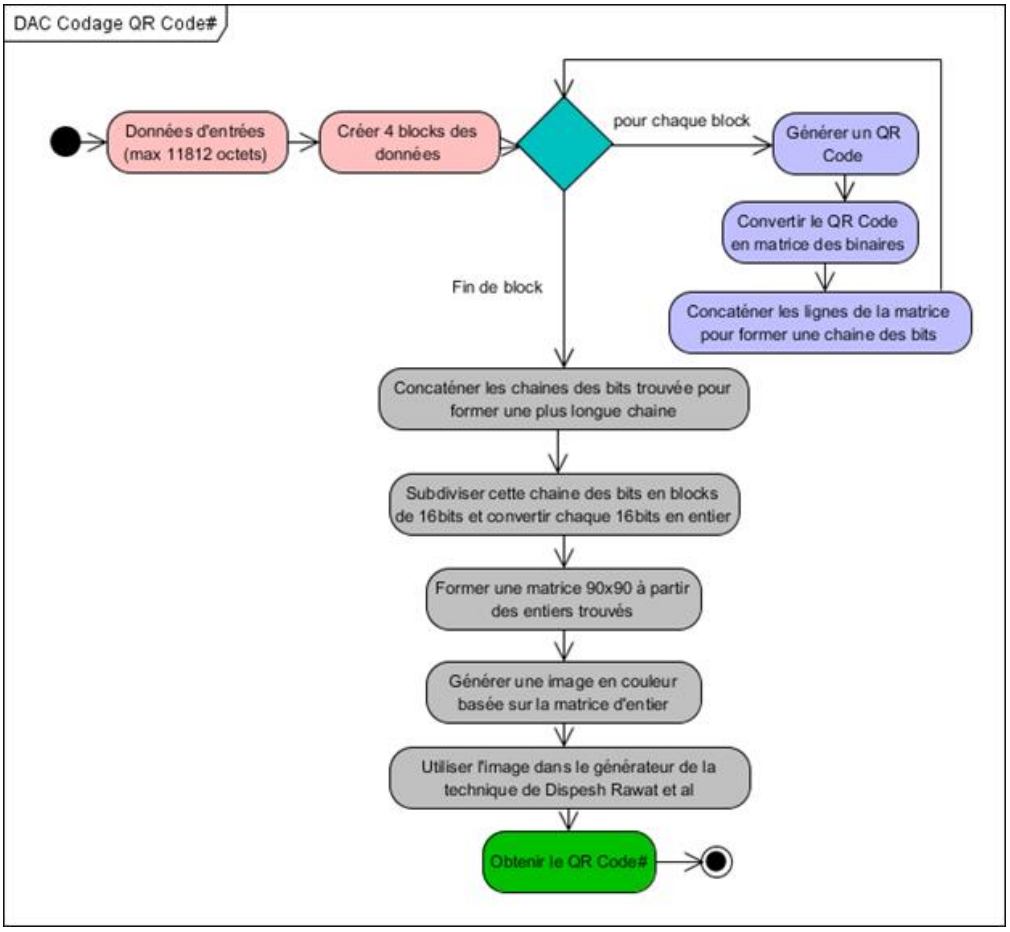
Dans cette section, nous présentons comment produire une image qui va servir à créer le QR Code#. Pour commencer, nous allons considérer un ensemble des données pouvant atteindre jusqu'à quatre fois la taille maximale d'un QR Code standard de version 40 et un niveau de correction d'erreur le plus bas (L), soit 7 089 chiffres, 4 296 Caractères alphanumériques, 2 953 Octets ou 23 648 bits. Ci-dessous sont les étapes à suivre :

1. Subdiviser en quatre blocs des données de même taille, les données d'entrées ;
2. Pour chaque bloc des données, créer un QR Code en utilisant la méthode de création décrite par le créateur du QR Code (Denso ADC, QR Code Essentiels, 2011) mais en personnalisant le résultat :
    a. Chaque module du QR Code est représenté par un pixel,
    b. La quit zone doit être de 3 modules au lieu de 4,
    c. Ce qui nous donne 4 QR Codes de dimension 180x180 pixels ou modules.
3. Chacun de ces QR Codes est converti en une matrice des pixels. Comme chaque pixel correspond à un module et que chaque module noir vaut 0 et le module blanc 1, nous formons dans ce cas une matrice carrée des 1 et des 0 avec comme dimensions 180x180 ;
4. Les lignes de chacune de 4 matrices créées sont alors concaténées pour former une seule ligne ;
5. Les bits de cette ligne sont à leurs tour concaténés pour former une chaine des bits ;

6. Cette chaine des bits est subdivisée en blocs de 16 bits et chaque 16 bits est converti en entier.
7. A partir de ces entiers, construire une nouvelle matrice d'entiers de dimension 90x90
8. Etant donné que dans une image, chaque pixel représentant une couleur donnée peut correspondre à un entier donné (CLAUDE, 2008), nous prenons notre matrice d'entier 90x90 et nous en faisons une matrice des pixels en couleur.
9. La matrice des pixels est convertie en une image de résolution 90x90 pixels.
10. En définitive, l'image créée, contenant à son sein quatre QR Codes de taille maximale est alors utilisée comme données d'entrées dans l'algorithme de Dipesh Rawat et al et ainsi produire notre QR Code#.

Le diagramme d'activité ci-dessous, illustre ce processus de génération du QR Code# :

Figure N°4 Processus de codage



Source : Auteur

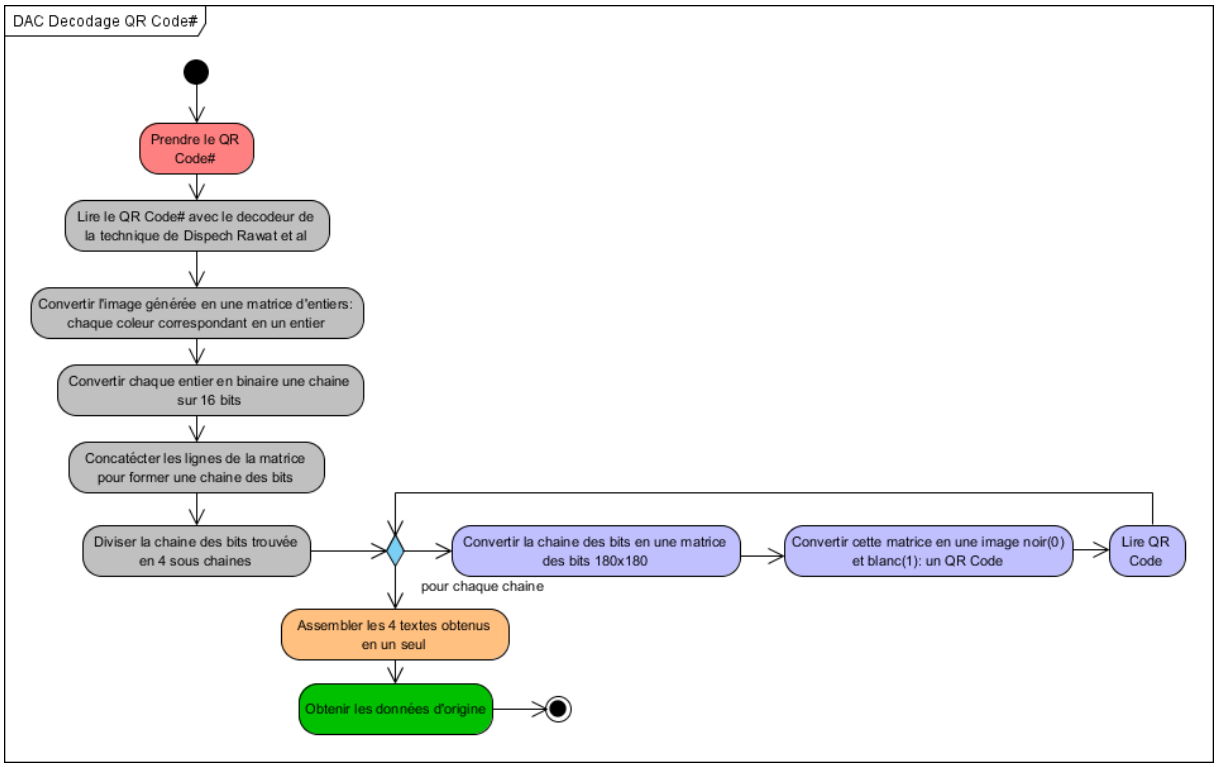
4. Processus de décodage du QR Code#

En prenant notre QR Code ++ que nous voulons décoder pour en extraire les données qui y sont cachées, nous pouvons passer par les étapes suivantes :

1. Lire le QR Code++ par le processus de décodage décrit par la technique de Dipesh Rawat et al (Rawat, Sahu, & Puthran, 2015)
2. L'image 90x90 en couleur générée est convertie en une matrice d'entiers, en exploitant chaque pixel de l'image ;
3. Chaque entier de la matrice est converti en une séquence de 16 bits. Si l'entier génère moins que 16 bits, on ajoute des 0 à gauche jusqu'à atteindre 16 bits ;
4. Toutes les lignes de cette matrice sont concaténées pour former une seule ligne constituée des blocks de 16 bits ;
5. Ces blocks de 16 bits sont ensuite concaténés pour former une seule chaine des bits ;
6. Cette chaine est subdivisée en 4 sous chaines et chacun convertie en une matrice des bits 180x180 ;
7. Chacun de ces 4 matrices, est convertie en matrice des pixels dont la valeur 1 correspond à la valeur d'un pixel de couleur blanche et la valeur 0 à un pixel de couleur noire ;
8. Chacun de ces matrices des pixels, est convertie en une image correspondant à un QR Code de version 40 et de bas niveau de correction d'erreur (L) ;
9. Chaque QR Code est enfin décodé par la technique standard (Denso ADC, QR Code Essentiels, 2011) et les données trouvées sont mise ensemble pour reconstituer les données d'origines qui représentent 4 fois les données qui devraient être encodées normalement.

Voici ci-après, le diagramme d'activités détaillant ce processus :

Figure N°5 processus de décodage du QR Code#



Source : Auteur

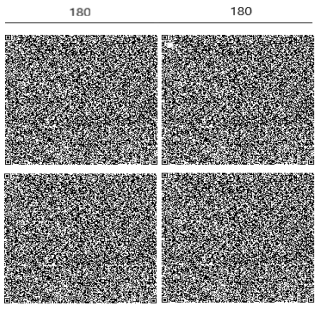
5. Implémentation du résultat

L'implémentation de résultat de cette étude a été faite en Java. Le choix de cette technologie a été guidé par plusieurs critères dont : facilité d'apprentissage, facilité de créer des interfaces utilisateurs multiplateformes, facilité de créer des applications pour smartphone.

Considérant un texte avec 2917 caractères en mode byte ou ECI, on peut générer un QR Code de version 40 et de niveau de correction d'erreur Low. Pour des raisons d'exemple nous avons quadruplet ces données pour en produire 4 QR Code.

Avec les modifications opérées dans le générateur du QR Code standard, nous obtenons des QR Codes de dimension 180x180 pixels comme le montre la figure suivante :

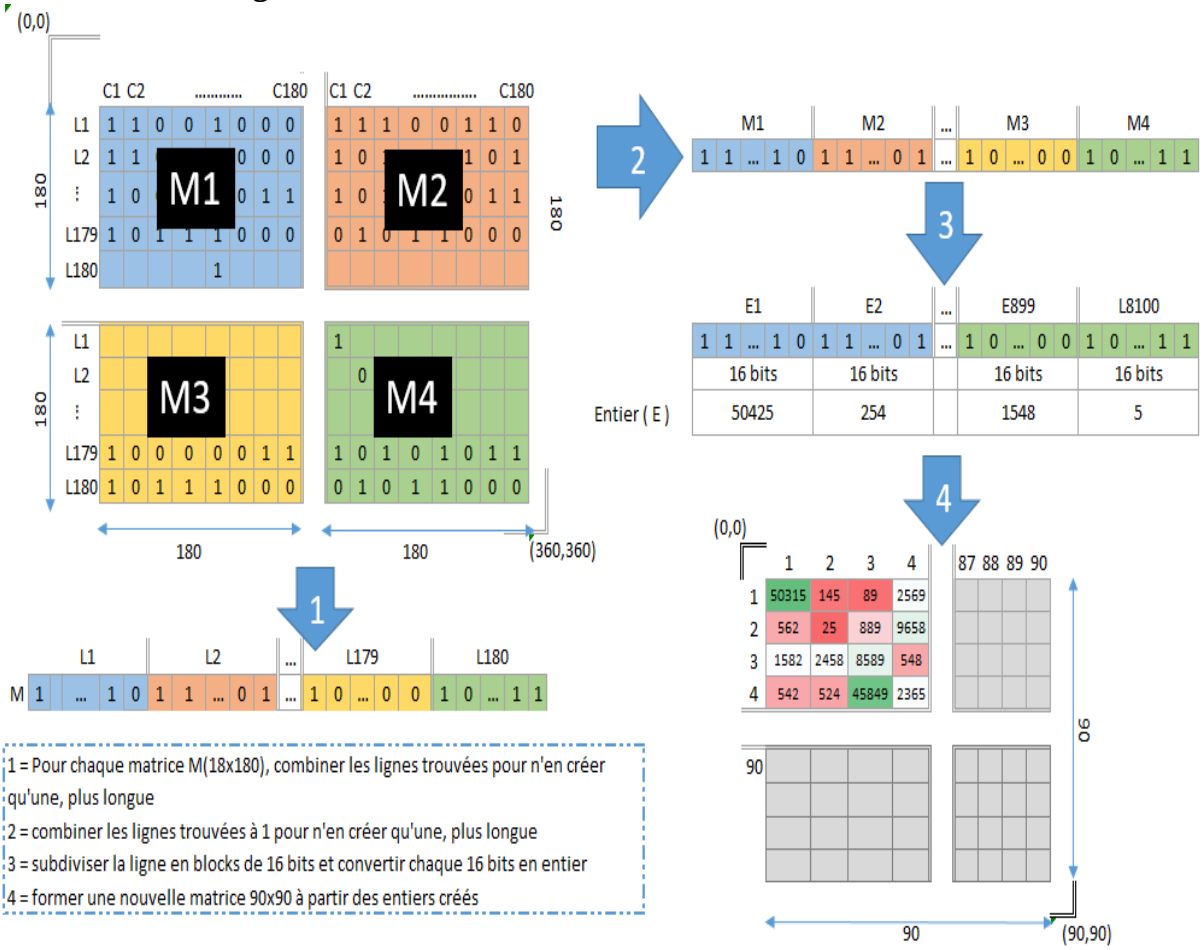
Figure N°6. 4 QR Codes standards générés



Source : Auteur

Par la suite, nous convertissons chacun de ces QR Codes en matrice des bits et les lignes de chaque matrice sont concaténées les unes à la suite des autres, puis toutes les lignes sont concaténées pour n'en former qu'une. En subdivisant cette ligne en blocks de 16 bits, en convertissant chaque 16 bits en entier et en créant une matrice d'entier à partir de ces blocks, nous obtenons une matrice de dimension 90x90. La figure suivante illustre ce procédé.

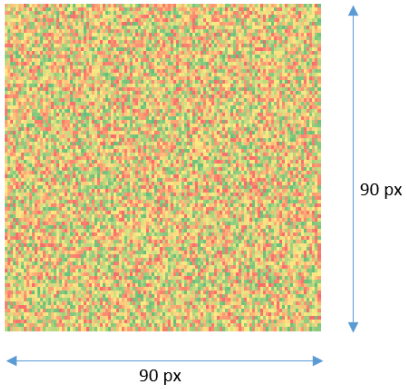
Figure N°7. Procédé de traitement des matrices des bits



Source : Auteur

Cette matrice d'entiers 90x90 est convertie en une matrice des pixels en couleur pour obtenir l'image de la figure qui suit :

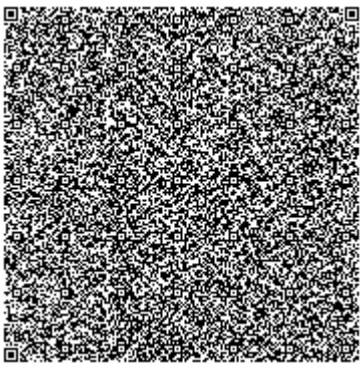
Figure N°8 : image 90x90 et représentant nos données



Source : Auteur

En utilisant cette image dans la technique de Dipesh Rawat et al (Rawat, Sahu, & Puthran, 2015) où elle compressée et cryptée avec le codage Base64, nous obtenons en définitive notre QR Code# comme le montre la figure suivante :

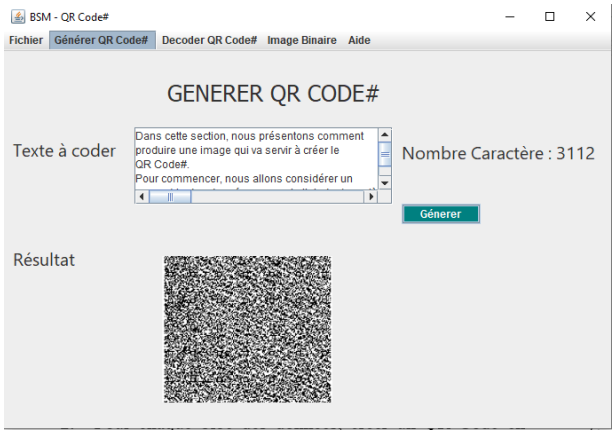
Figure N°9 : QR Code# pour l'image 90x90



Source : **Auteur**

La figure ci-dessous présente l'interface Homme-machine de notre prototype de génération du QR Code# que nous avons développé en java :

Figure N°10. IHM Générateur QR Code#



Source : Auteur

Conclusion

Cet article met en exergue l'optimisation de la capacité de stockage d'un QR code standard et sécurité de données dont l'objectif principal était de proposer et implémenter un algorithme pouvant garantir la sécurité des données du QR Code et améliorer sa capacité de stockage en quadruplant les données d'entrées.

Pour parvenir au résultat escompté, nous sommes partis d'abord d'une revue de la littérature qui nous a permis de comprendre le principe de fonctionnement du QR Code, son processus de création et de lecture. Ensuite nous avons parcouru un ensemble des travaux en relation avec notre thème de recherche.

Ayant étudié minutieusement l'un de ces travaux, (Rawat, Sahu, & Puthran, 2015) , nous sommes arrivés à mettre en place une technique permettant de cacher les informations à encoder dans un QR Code et de quadrupler les données à encoder par rapport aux données maximales que stocker un QR Code standard.

Le résultat de cette technique se place en amont de la technique de Dipesh Rawat et al (Rawat, Sahu, & Puthran, 2015) et dont le résultat produit un QR Code que nous qualifions de QR Code#.

En comparant le résultat de notre technique avec ceux des travaux précédents, nous pouvons affirmer nos objectifs. Premièrement, notre technique sécurise au mieux les données encodées dans le QR code vu les procédé de dissimilation des données que nous avons appliqués à trois niveau du processus de génération de notre QR Code#. Deuxièmement, la taille des données stockées dans le QR Code# devient de loin supérieure à la taille maximale des données que peut

stocker un QR Code standard à sa dernière version (40). Ce qui dépasse les autres recherches à notre connaissance sur le même sujet.

L'implémentation de notre technique a été faite en Java WindowBuilder. La compression et cryptage des données sont faites avec le codage Base64 (Josefsson, Octobre 2006) .

Limites

Se basant sur la définition même du QR Code (Denso ADC, QR Code Essentiels, 2011) : Code à accès rapide, notre technique respecte mal cette définition. Le processus de génération et lecture de notre QR Code# étant très complexe, le temps de création et de lecture de notre QR Code# sera importante (un point non aborder dans notre étude).

Perspectives

Notre souhait serait qu'à l'avenir notre technique soit optimisée pour que la génération et la lecture de notre QR Code++ se fasse le plus rapidement possible conformément au standard défini par le créateur du QR Code.

ANNEXES

Tableau N°3. Tableau de capacité de QR Code

Version	Modules	Error Correction	Data bits	Numeric	Alphanumeric	Binary
1	21x21	L	152	41	25	17
		M	128	34	20	14
		Q	104	27	16	11
		H	72	17	10	7
2	25x25	L	272	77	47	32
		M	224	63	38	26
		Q	176	48	29	20
		H	128	34	20	14
3	29x29	L	440	127	77	53
		M	352	101	61	42
		Q	272	77	47	32
		H	208	58	35	24
4	33x33	L	640	187	114	78
		M	512	149	90	62
		Q	384	111	67	46
		H	288	82	50	34
5	37x37	L	864	255	154	106
		M	688	202	122	84
		Q	496	144	87	60
		H	368	106	64	44
6	41x41	L	1.088	322	195	134
		M	864	255	154	106
		Q	608	178	108	74
		H	480	139	84	58
7	45x45	L	1.248	370	224	154
		M	992	293	178	122
		Q	704	207	125	86
		H	528	154	93	64
8	49x49	L	1.552	461	279	192
		M	1.232	365	221	152
		Q	880	259	157	108
		H	688	202	122	84
9	53x53	L	1.856	552	335	230
		M	1.456	432	262	180
		Q	1.056	312	189	130
		H	800	235	143	98
10	57x57	L	2.192	652	395	271
		M	1.728	513	311	213
		Q	1.232	364	221	151
		H	976	288	174	119
11	61x61	L	2.592	772	468	321
		M	2.032	604	366	251

		Q	1.440	427	259	177
		H	1.120	331	200	137
12	65x65	L	2.960	883	535	367
		M	2.320	691	419	287
		Q	1.648	489	296	203
		H	1.264	374	227	155
13	69x69	L	3,424	1,022	619	425
		M	2.672	796	483	331
		Q	1.952	580	352	241
		H	1.440	427	259	177
14	73x73	L	3.688	1.101	667	458
		M	2.920	871	528	362
		Q	2.088	621	376	258
		H	1.576	468	283	194
15	77x77	L	4.184	1.250	758	520
		M	3.320	991	600	412
		Q	2.360	703	426	292
		H	1.784	530	321	220
16	81x81	L	4.712	1.408	854	586
		M	3.624	1.082	656	450
		Q	2.600	775	470	322
		H	2.024	602	365	250
17	85x85	L	5.176	1.548	938	644
		M	4.056	1.212	734	504
		Q	2.936	876	531	364
		H	2.264	674	408	280
18	89x89	L	5.768	1.725	1.046	718
		M	4.504	1.346	816	560
		Q	3.176	948	574	394
		H	2.504	746	452	310
19	93x93	L	6.360	1.903	1.153	792
		M	5.016	1.500	909	624
		Q	3.560	1.063	644	442
		H	2.728	813	493	338
20	97x97	L	6.888	2.061	1.249	858
		M	5.352	1.600	970	666
		Q	3.880	1.159	702	482
		H	3.080	919	557	382
21	101x101	L	7.456	2.232	1.352	929
		M	5.712	1.708	1.035	711
		Q	4.096	1.224	742	509
		H	3.248	969	587	403
22	105x105	L	8.048	2.409	1.460	1.003
		M	6.256	1.872	1.134	779
		Q	4.544	1.358	823	565
		H	3.536	1.056	640	439
23	109x109	L	8.752	2.620	1.588	1.091

		M	6.880	2.059	1.248	857
		Q	4.912	1.468	890	611
		H	3.712	1.108	672	461
24	113x113	L	9.392	2.812	1.704	1.171
		M	7.312	2.188	1.326	911
		Q	5.312	1.588	963	661
		H	4.112	1.228	744	511
25	117x117	L	10.208	3.057	1.853	1.273
		M	8.000	2.395	1.451	997
		Q	5.744	1.718	1.041	715
		H	4.304	1.286	779	535
26	121x121	L	10.960	3.283	1.990	1.367
		M	8.496	2.544	1.542	1.,059
		Q	6.032	1.804	1.094	751
		H	4.768	1.425	864	593
27	125x125	L	11.744	3.514	2.132	1.465
		M	9.024	2.701	1.637	1,125
		Q	6.464	1.933	1.172	805
		H	5.024	1.501	910	625
28	129x129	L	12.248	3.669	2.223	1,528
		M	9.544	2.857	1.732	1.190
		Q	6.968	2.085	1.263	868
		H	5.288	1.581	958	658
29	133x133	L	13.048	3.909	2.369	1.628
		M	10.136	3.035	1.839	1.264
		Q	7.288	2.181	1.322	908
		H	5.608	1.677	1.016	698
30	137x137	L	13.880	4.158	2.520	1,732
		M	10.984	3.289	1.994	1.370
		Q	7.880	2.358	1.429	982
		H	5.960	1.782	1.080	742
31	141x141	L	14.744	4.417	2.677	1.840
		M	11.640	3.486	2.113	1.452
		Q	8.264	2.473	1.499	1.030
		H	6.344	1.897	1.150	790
32	145x145	L	15.640	4.686	2.840	1,952
		M	12.328	3.693	2.238	1.538
		Q	8.920	2.670	1.618	1.112
		H	6.760	2.022	1.226	842
33	149x149	L	16.568	4.965	3.009	2.068
		M	13.048	3.909	2.369	1.628
		Q	9.368	2.805	1.700	1.168
		H	7.208	2.157	1.307	898
34	153x153	L	17.528	5.253	3.183	2.188
		M	13.800	4.134	2.506	1,722
		Q	9.848	2.949	1.787	1,228
		H	7.688	2.301	1.394	958

35	157x157	L	18.448	5.529	3.351	2.303
		M	14.496	4.343	2.632	1.809
		Q	10.288	3.081	1.867	1.283
		H	7,888	2.361	1.431	983
36	161x161	L	19.472	5.836	3.537	2.431
		M	15.312	4.588	2.780	1.911
		Q	10.832	3.244	1.966	1.351
		H	8.432	2.524	1.530	1.051
37	165x165	L	20.528	6.153	3.729	2.563
		M	15.936	4.775	2.894	1.989
		Q	11.408	3.417	2.071	1.423
		H	8.768	2.625	1.591	1.093
38	169x169	L	21.616	6.479	3.927	2.699
		M	16.816	5.039	3.054	2.099
		Q	12.016	3.599	2.181	1.499
		H	9.136	2.735	1.658	1,139
39	173x173	L	22.496	6.743	4.087	2.809
		M	17.728	5.313	3.220	2.213
		Q	12.656	3.791	2.298	1.579
		H	9.776	2,927	1.774	1.219
40	177x177	L	23.648	7.089	4.296	2.953
		M	18.672	5.596	3.391	2,331
		Q	13.328	3.993	2,420	1.663
		H	10.208	3,057	1.852	1.273

Source : (Rawat, Sahu, & Puthran, 2015)

BIBLIOGRAPHIE

Brackins, J., & Qiao, M. (s.d.). A Secure QR Code Scheme. Department of Mathematics and Computer Science, South Dakota School of Mines and Technology.

CLAUDE, D. (2008). *Programmer en Java.* Paris: Eyroles.

Denso ADC. (2011). *QR Code Essentiels.* Consulté le Aout 5, 2022, sur http://www.nacs.org/LinkClick.aspx?fileticket=D1FpVAvvJuo%3D&tabid=1426&mid

Denso ADC. (2011). *QR Code Essentiels.* Consulté le 04 2017, sur http://www.nacs.org/LinkClick.aspx?fileticket=D1FpVAvvJuo%3D&tabid=1426&mid

Denso Wave, i. (s.d.). *History of QR Code.* Consulté le aout 15, 2022, sur qrcode: <http://www.qrcode.com/en/history/>

Goel, S., & Singh, A. K. (Sep 2014). Cost minimization by QR Code compression. *15*(4).

Josefsson, S. (Octobre 2006). The Base16, Base32, and Base64 Data Encodings. *RFC 4648*.

Lu, J., Yang, Z., Li, L., Yuan, W., Li, L., & Chang, C.-C. (2017). Multiple Schemes for Mobile Payment Authentication Using QR Code and Visual Cryptography. *2017*(4356038).

Narayanan, A. S. (2012). QR Codes and Security Solutions. *3*(7).

Peng, K., Sanabria, H., Wu, D., & Zhu, C. (s.d.). Security Overview of QR Codes.

Petrova, K., Romanello, A., B. Dawn, M., & Sandra, A. V. (2016). QR Codes Advantages and Dangers. *2.* In Proceedings of the 13th International Joint Conference on e-Business and Telecommunications (ICETE 2016) .

Prathibha, N. P., & Naresh, K. (July 2014). Improving the capacity of QR Code by using color technique. *3*(7).

Rawat, D., Sahu, R., & Puthran, Y. (2015). Optimizing the Capacity of QR Code To Store Encrypted Image. *3*(1).

Rikala, J., & Kankaanranta, M. (s.d.). The Use of Quick Response Codes in the Classroom.

Singh, A., & Singh, D. P. (June 2016). A REVIEW: QR CODES AND ITS IMAGE PRE-PROCESSING METHOD. *5*(6).

Skokov, O., Sneps-Sneppe, M., & Namiot, D. (s.d.). Context-aware QR-codes.

Victor, N. (12-2012). Enhancing the data capacity of QR Codes by compressing the data before generation. *60*(2).

Warasart, M., & Kuacharoen, P. (2012). Paper-based Document Authentication using Digital Signature and QR Codes. Singapore: International Conference on Computer Engineering and Technology.