

# SCIENTIFIC REPORTS



OPEN

## Quantum Hash function and its application to privacy amplification in quantum key distribution, pseudo-random number generation and image encryption

Received: 05 December 2014

Accepted: 02 November 2015

Published: 29 January 2016

Yu-Guang Yang<sup>1,2,3,4</sup>, Peng Xu<sup>1</sup>, Rui Yang<sup>1</sup>, Yi-Hua Zhou<sup>1</sup> & Wei-Min Shi<sup>1</sup>

Quantum information and quantum computation have achieved a huge success during the last years. In this paper, we investigate the capability of quantum Hash function, which can be constructed by subtly modifying quantum walks, a famous quantum computation model. It is found that quantum Hash function can act as a hash function for the privacy amplification process of quantum key distribution systems with higher security. As a byproduct, quantum Hash function can also be used for pseudo-random number generation due to its inherent chaotic dynamics. Further we discuss the application of quantum Hash function to image encryption and propose a novel image encryption algorithm. Numerical simulations and performance comparisons show that quantum Hash function is eligible for privacy amplification in quantum key distribution, pseudo-random number generation and image encryption in terms of various hash tests and randomness tests. It extends the scope of application of quantum computation and quantum information.

With the rapid development of quantum communication, quantum key distribution (QKD) is the most mature branch of quantum communication. The goal of QKD is to create an absolutely secure key between two communicating parties. QKD generally contains three steps: (1) raw key sifting, (2) error reconciliation, and (3) privacy amplification. As an important step of QKD, the privacy amplification process is implemented by adopting universal hash functions<sup>1</sup>. However, these hash functions are generally constructed based on mathematics complexity and thus they are computationally secure. Because the fundamental principles of quantum mechanics ensure lots of quantum cryptographic protocols<sup>2-4</sup> with unconditional security, this stimulates us to consider the privacy amplification problem in the context of quantum information, and intend to get a more secure solution to the privacy amplification process.

In this paper, we construct a quantum Hash function (QHF) by subtly modifying the quantum walks (QW) model<sup>5-13</sup> and it can be used for the privacy amplification process of QKD systems with higher security by means of the physical principles of quantum mechanics. As a byproduct, QHF can also be used for pseudo-random number generation due to its inherent chaotic dynamics and further we propose a novel QHF-based image encryption algorithm. Numerical simulations and performance comparisons show that QHF is eligible as a hash function for privacy amplification in QKD, pseudo-random number generation and image encryption in terms of various hash tests and randomness tests.

Compared to the QW-based algorithm<sup>14</sup>, the novelty of the present QHF-based scheme lies in that the constructed QHF can be not only used for pseudo-random number generation and further image encryption, but also used for the privacy amplification process of QKD systems with higher security. It extends the scope of application of quantum computation and quantum information.

<sup>1</sup>College of Computer Science and Technology, Beijing University of Technology, Beijing 100124, China. <sup>2</sup>State Key Laboratory of Information Security (Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China. <sup>3</sup>Beijing Key Laboratory of Trusted Computing, Beijing 100124, China. <sup>4</sup>National Engineering Laboratory for Critical Technologies of Information Security Classified Protection, Beijing 100124, China. Correspondence and requests for materials should be addressed to Y.-G.Y. (email: yangyang7357@bjut.edu.cn)

## Results

**The construction of QHF.** QHF can be constructed by subtly modifying QW. QW has two models: discrete QW and continuous QW<sup>5</sup>. The basic discrete QW includes two quantum systems: walker and coin. The state of the walker-coin system is denoted by a vector in the Hilbert space  $H_t = H_p \otimes H_c$ , where the subscripts  $p$  and  $c$  stand for the walker and the coin, respectively. The motion of the walk is conditioned by the coin state via a conditional shift operator

$$\hat{S} = \sum_x (|x+1, 0\rangle\langle x, 0| + |x-1, 1\rangle\langle x, 1|), \quad (1)$$

where the summation symbol denotes the sum over all possible positions. The evolution of the total quantum system can be implemented by repeating the global unitary operator

$$\hat{U} = \hat{S}(\hat{I} \otimes \hat{C}), \quad (2)$$

where  $\hat{I}$  is the identity operator and  $\hat{C}$  is the coin operator applied on the coin state. Hence the final state  $|\psi\rangle_t$  after  $t$  steps is expressed by

$$|\psi\rangle_t = (\hat{U})^t |\psi\rangle_0 = \sum_x \sum_v \lambda_{x,v} |x, v\rangle, \quad (3)$$

and the probability of locating the walker at position  $x$  after  $t$  steps is

$$P(x, t) = \sum_{v \in \{0,1\}} |\langle x, v | (\hat{U})^t |\psi\rangle_{initial}|^2, \quad (4)$$

where  $|\psi\rangle_{initial}$  is the initial state of the total quantum system.

In a discrete-time QW, the coin operator is fixed. The resulting probability distribution relies on only the initial coin state and the step number. Suppose the coin operator at each step depends on a binary string, i.e., *message*, and accordingly a QHF is constructed, similar to that in Ref.15. The input of the constructed QHF is a binary string, i.e., *message* and the resulting probability distribution is used as the output hash value. The coin state is the control parameter so the constructed QHF is a keyed one. The  $n$ th bit of the *message* controls the  $n$ th step of the walk. Here we introduce two coin operators, i.e., the Grover operator  $\hat{C}_0$ <sup>16</sup> and the coin operator  $\hat{C}_1$ <sup>17</sup> in equation (5) and equation (6) respectively. The *message* bit “0” denotes  $\hat{C}_0$  and “1” for  $\hat{C}_1$ .

$$\hat{C}_0 = \frac{1}{2} \begin{pmatrix} -1 & 1 & 1 & 1 \\ 1 & -1 & 1 & 1 \\ 1 & 1 & -1 & 1 \\ 1 & 1 & 1 & -1 \end{pmatrix}, \quad (5)$$

$$\hat{C}_1 = \frac{1}{2} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & -1 & 1 \\ -1 & 1 & -1 & 1 \\ -1 & -1 & 1 & 1 \end{pmatrix}. \quad (6)$$

The construction process of the QHF is as follows:

- (1) Select the parameters  $(n, (\alpha, \beta, \chi, \delta))$  and the *message* with arbitrary length.
- (2) Run the one-dimensional two-particle discrete-time QW on a circle under the control of the *message* and generate the output hash value, i.e., the probability distribution. Here  $\alpha, \beta, \chi, \delta$  are the amplitudes of the initial coin state  $|v, \tau\rangle = (\alpha|00\rangle + \beta|01\rangle + \chi|10\rangle + \delta|11\rangle)$ .  $n$  is the node number of a circle.
- (3) Multiply all values in the resulting probability distribution by  $10^8$  modulo 256 to form a binary string as a secret key  $K$ , i.e., the hash value.

**The hash property of the proposed QHF.** In this section, we performed several hash tests to evaluate the performance of the proposed QHF. The *message* can be randomly chosen as shown in Supplemental materials.

**Statistical distribution of hash value.** Based on the principles of quantum mechanics, the security of the QHF can be proved partly by the uniform distribution of the hash value. The plots of the ASCII codes of the *message* and its hash value are shown in Supplementary Fig. S1 online. Supplementary Fig. S1(a) demonstrates that the ASCII code of the *message* is located within a small range, but in Supplementary Fig. S1(b), the hash value of the *message* in hexadecimal format is scattered uniformly.

**Sensitivity of hash value to message.** C1, C2, C3 and C4 represent the *message*, and the *message* with tiny modifications respectively. The results listed below show the high sensitivity to the *message* and the tiny changes.

- Condition 1: The original *message*;
- Condition 2: Change the 8<sup>th</sup> bit from 0 to 1;
- Condition 3: Delete the last bit of the *message*;

	$N=1024$	$N=2048$	$N=10,000$	Mean
$\bar{B}$	63.5654	63.5864	64.2894	63.8137
$P(\%)$	49.6605	49.6769	50.2261	49.8545
$\Delta B$	5.4616	5.5841	5.6686	5.6314
$\Delta P$	4.3881	4.3626	4.4286	4.3931
$B_{\min}$	45	44	43	44
$B_{\max}$	81	83	89	84.3333

**Table 1. The static number of changed bit  $B_i$ .** The mean changed bit number  $B$  and the mean changed probability  $P$  are very close to the ideal value 64 bit and 50% respectively.  $\Delta B$  and  $\Delta P$  are very little, so that it demonstrates the stability of diffusion and confusion. The excellent statistical effect ensures that it is impossible to forge plaintext-cipher text pairs given several known plaintext-cipher text pairs.

Condition 4: Insert a bit in front of the 100<sup>th</sup> bit.

The corresponding 128-bit hash value in the hexadecimal format is given by:

Condition 1: 8AE72983687E9AD1B6FCA54546AE7799;

Condition 2: 3BD58DB7B86827AE6323E6E496A634A8;

Condition 3: E9678E1EA9180A8EE01AA008EB46E989;

Condition 4: E9277B78B3D62DEB77839DB9E90F210D.

The plots of the hash values are shown respectively in Supplementary Fig. S2 online and it is clearly indicated that any tiny modification to the *message* or the key will cause a substantial change in the final hash value.

**Statistical analysis of diffusion and confusion.** The diffusion and confusion tests are performed as follows:

- (1) Select a *message* and generate the corresponding hash value.
- (2) Change one bit of the *message* randomly and generate a new hash value.
- (3) Compare the two hash values and count the changed bits called  $B_i$ .
- (4) Repeat steps (1) to (3)  $N$  times.

The corresponding distribution and the histogram of  $B_i$  are shown respectively in Supplementary Figs S3(a) and S3(b) online, where  $N=10,000$ .

Minimum changed bit number  $B_{\min} = \min(\{B_i\}_1^N)$ ;

Maximum changed bit number  $B_{\max} = \max(\{B_i\}_1^N)$ ;

Mean changed bit number  $\bar{B} = \sum_{i=1}^N B_i / N$ ;

Mean changed probability  $P = (\bar{B}/128) \times 100\%$ ;

Standard variance of the changed bit number  $\Delta B = \sqrt{\frac{1}{N-1} \sum_{i=1}^N (B_i - \bar{B})^2}$ ;

Standard variance of the changed probability  $\Delta P = \sqrt{\frac{1}{N-1} \sum_{i=1}^N (B_i/128 - P)^2} \times 100\%$ .

Next the diffusion and confusion tests are performed with  $N=1024, 2048, 10,000$ , respectively, as shown in Table 1. We concluded from the tests that the mean changed bit number  $B$  and the mean changed probability  $P$  are close to the ideal value 64 and 50% respectively.  $\Delta B$  and  $\Delta P$  are very little, so that it demonstrates the stability of diffusion and confusion. The excellent statistical effect ensures that it is impossible to forge plaintext-cipher pairs given known plaintext-cipher pairs.

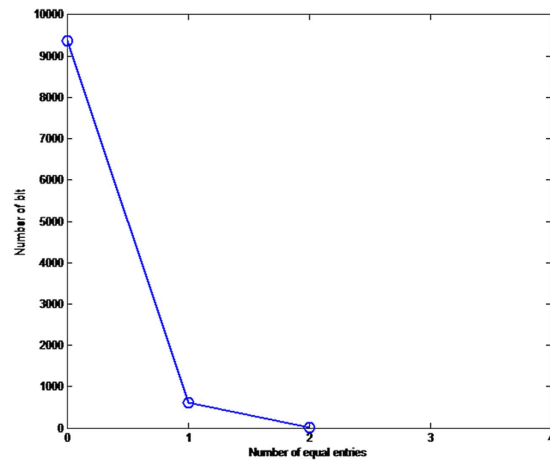
**Collision analysis.** It is hard to provide a mathematical proof on the capability of collision resistance of chaotic hash functions. Thus, we performed the following test for collision resistance:

- (1) Select an original message randomly and generate the corresponding hash value in ASCII format.
- (2) Choose a bit in the message randomly and change its value.
- (3) Generate a new hash value.
- (4) Compare these two hash values and count the number of ASCII characters with the same value at the same location.

Moreover, the absolute difference of the two hash values, i.e.,  $d$ , and the theoretical number of  $\omega$  with different values through  $N$  independent tests, i.e.,  $W_N(\omega)$  can be computed according to the following formulas:

$$d = \sum_{i=1}^N |t(e_i) - t(e'_i)| \quad (7)$$

$$\omega = \sum_{i=1}^N f(t(e_i) - t(e'_i)), \quad \text{where } f(x) = \begin{cases} 1 & x = y, \\ 0 & x \neq y. \end{cases} \quad (8)$$



**Figure 1.** Distribution of the number of positions where the ASCII characters are identical in the 128-bit hash values generated for test, where  $N=10,000$ . (see text in the section entitled Results).

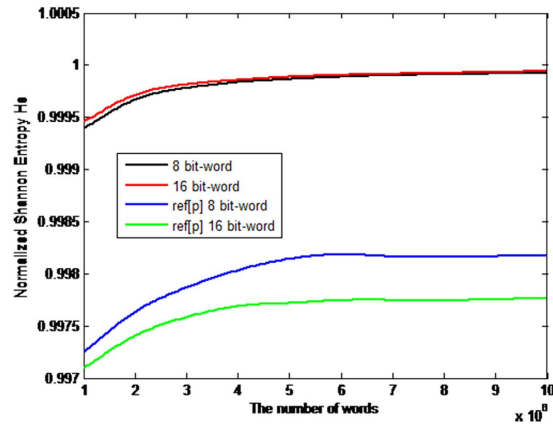
$$W_N(\omega) = N \times Prob\{\omega\} = N \frac{s!}{\omega!(s-\omega)!} \left(\frac{1}{2^k}\right)^\omega \left(1 - \frac{1}{2^k}\right)^{s-\omega}, \quad (9)$$

where  $e_i$  and  $e'_i$  are the  $i$ th entries of the original and new hash values, respectively. The function  $t(\cdot)$  converts the entries to their equivalent decimal value. We run this test  $N=10,000$  times, and listed the maximum, minimum, and mean of  $d$  in Supplementary Table 1 online respectively. In equation (8),  $\omega=0, 1, \dots, s$ . The experimental values of  $W_N(\omega)$  in the proposed scheme are:  $W_N(0)=9367$ ,  $W_N(1)=617$ ,  $W_N(2)=16$ , and  $W_N(\omega)=0$  for  $\omega=3, 4, \dots, 16$  respectively. The distribution of the number of ASCII characters with the same value at the same location in the hash value is displayed in Fig. 1.

**Uniform distribution on hash space.** In order to check the distribution capacity in hash space, similar to that in ref. 18, we generated two hash values according to the method described in previous subsection and then counted the number of the changed bits at each location. The minimum, maximum and mean of changed bit numbers are 4072, 5689 and 4973.5, respectively for  $N=10,000$ . The statistical results for  $N=10,000$  are shown in Supplementary Fig. S4 online. The mean of the changed bit number 4973.5 is very close to the ideal value 5000, which accounts for half of the test times. It can be concluded that the hash value is distributed uniformly in the hash space as all the changed bit numbers are around the ideal value. Obviously, this demonstrates the resistance against statistical attack.

**Resistance to birthday attack.** Similar to collision resistance, the birthday attack is mainly aimed to find two messages with identical hash values with less than  $2^{n/2}$  trials ( $n$  is the size of hash value). According to the current computing power, the size of hash value should be greater than 128 to ensure the birthday attack complexity is greater than 264. In our proposed hash scheme, the length of hash value is 128-bit and it can be easily extended to 256-bit or 512-bit. So the attack difficulty is at least  $2^{64}$ , which is huge enough to resist brute force attack and birthday attack. Also in the proposed algorithm, in order to analyze the security, several tests including the SP800-22 test and the collision tests were applied. Therefore, the results of the tests, the size of the hash value, and the collision resistance of the proposed algorithm suggest that the birthday attack is almost impossible and that the proposed algorithm is resistant against this type of attack.

**Speed analysis.** Our proposed hash algorithm does not need to pad bits. The time required to generate a hash value is closely related to the length of the message. The algorithm is simulated in MATLAB on a PC with Intel(R) Core(TM) i3-2370M CPU 2.40 GHz 2 GB RAM running on Windows 7 professional OS. The average speed of our algorithm is 0.2 Mbit/s. Although the speed of the proposed algorithm is lower than the traditional hash functions such as SHA-1 and MD5<sup>19</sup>, it is acceptable for practical use. At the same time, the algorithm is so flexible since the length of hash value can be 128, 256, or 512 bits. Moreover, because of the excellent properties of quantum parallel computing, the speed of the proposed algorithm will increase exponentially in the quantum computing environment. For example, finding the prime factorization of an  $n$ -bit integer is thought to require  $\exp(\Theta(n^{1/3} \log^{2/3} n))$  operations using the best classical algorithm. In contrast, a quantum algorithm can accomplish the same task using  $\Theta(n^2 \log n \log \log n)$  operations<sup>20</sup>. That is, a quantum computer can factor a number exponentially faster than the best known classical algorithms.



**Figure 2. Comparisons in terms of Normalized Shannon entropy  $H_S$ .** The black and red curves denote the Normalized Shannon entropy  $H_S$  for the number of 8 bits and 16 bits-words respectively in our proposal, while the blue and green curves represent the Normalized Shannon entropy of the QW-based scheme<sup>14</sup> for the number of 8 bits and 16 bits-words respectively. (see text in the section entitled Results).

**Flexibility.** Our hash scheme is based on the QW model. Although the proposed hash function is constructed as a keyed one, we can also regard it as an unkeyed one if the initial parameters of the coin state of QHF act as the constants. Besides we can also get a 256-bit or 512-bit hash function with a slight change in the original version, similar to refs 21,22.

**Resistance to meet-in-the-middle attack.** The meet-in-the-middle attack is valid for the block cipher encryption mode. Block cipher mode allows the use of a cipher key for encrypting more than one block of data. In contrast, in our proposed algorithm, we use just the nonlinear quantum map to construct the hash function, so the attack is useless for the proposed algorithm.

**Resistance to forgery attack.** Most of the parallel hash function algorithms have a mixing section in their structure which usually uses the XOR operation for preventing forgery attack. Unfortunately, some of these algorithms are broken by such an attack<sup>23,24</sup>. In the proposed algorithm, the state evolution of the total quantum system can be implemented by repeating the sequence of the coin flipping operator and the conditional shift operator step by step according to the message (so-called discrete time). That is, the  $n$ th bit of the message controls the  $n$ th step of the walk, i.e., applying the Grover operator  $\hat{C}_0$ <sup>16</sup> or the coin operator  $\hat{C}_1$ <sup>17</sup> on the coin state. This leads to high complexity in mixture and can resist forgery attack in any section of the proposed algorithm.

**Security analyses of the QHF-based pseudo-random number generator (PRNG).** QHF can also be used for pseudo-random number generation due to its inherent chaotic dynamics. To analyze the pseudo-randomness of the QHF-based PRNG, we analyzed its statistical properties and some quantifiers were proposed. The quantifiers are mainly classified into two classes: (i) quantifiers based on information theory<sup>25–27</sup>, (ii) quantifiers based on recurrence plots<sup>28,29</sup>.

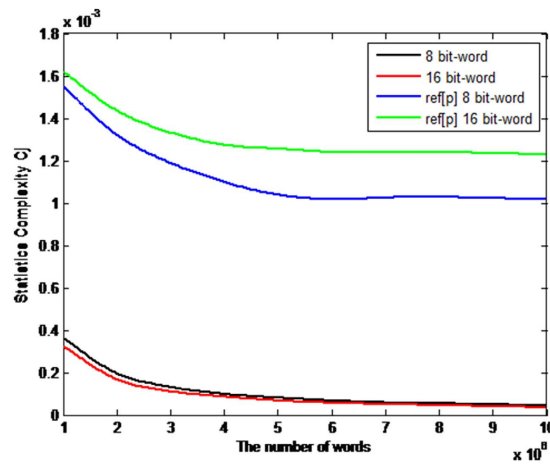
**Statistical complexity measure.** Complexity is a measure of off-equilibrium ‘order’. Statistical complexity measures (SCM) were proposed as quantifiers of the degree of physical structure in a signal<sup>25,30,31</sup>. Based on the method of Ref. 32, we analyzed the statistical complexity of the QHF-based PRNG. The intensive SCM ( $C_J[P]$ ) can be considered as a quantity that characterizes the probability distribution  $P$  associated with the time series generated by the dynamical system<sup>32</sup>. It quantifies not only randomness but also the presence of correlational structures<sup>31,32</sup>. The measure of statistical complexity  $C_J[P]$  is defined as<sup>32</sup>:

$$C_J[P] = Q_J[P, P_e] \cdot H_S[P], \quad (10)$$

where the normalized entropic measure  $H_S[P] = S[P]/S_{\max}$  is associated with the probability distribution  $P$ , with  $S_{\max} = S[P_e]$  ( $0 \leq H_S \leq 1$ ) for the equilibrium distribution  $P_e$  and  $S$  is the Shannon entropy. The disequilibrium  $Q_J$  is defined in terms of the Jensen-Shannon divergence<sup>26,32</sup> by

$$Q_J[P, P_e] = Q_0 \{S[(P + P_e)/2] - S[P]/2 - S[P_e]/2\}, \quad (11)$$

with  $Q_0$  being the normalization constant ( $0 \leq Q_J \leq 1$ ). Thus, the disequilibrium  $Q_J$  is an intensive quantity. Following the methodology proposed by Bandt and Pompe<sup>33</sup>, the comparisons between our proposal and the QW-based PRNG<sup>14</sup> in terms of the normalized entropy  $H_S$  and the intensive statistical complexity  $C_J$  as functions of the number of 8 bits and 16 bits-words are shown in Figs 2 and 3 respectively. When the number of words of the analyzed sequence increases, the statistical complexity and the normalized entropy tend to 0 and 1 respectively. It



**Figure 3. Comparisons in terms of intensive statistical complexity measure  $C_j$  respectively.** The black and red curves denote the intensive statistical complexity measure  $C_j$  for the number of 8 bits and 16 bits-words respectively in our proposal, while the blue and green curves represent the intensive statistical complexity measure  $C_j$  of the QW-based scheme<sup>14</sup> for the number of 8 bits and 16 bits-words respectively. (see text in the section entitled Results).

is shown that given the same words, our scheme has better statistical complexity and normalized entropy than the QW-based PRNG<sup>14</sup>.

**Recurrence plots.** Recurrence is a fundamental property of dynamical systems, which can be exploited to characterize the system’s behavior in phase space. In 1987, Eckmann *et al.* introduced a powerful tool for visualization and analysis of recurrences called recurrence plot (RP)<sup>28</sup>. RP is a two-dimensional representation in which both axes are time ones. The recurrence of a state appearing at two given times  $t_i, t_j$  is pictured in the two-dimensional graph by means of a black dot.

To visualize the recurrences of states of a dynamical system, the RP of a trajectory  $\vec{x}_i \in \mathfrak{R}^d$  can be formally expressed by the matrix

$$R_{i,j}(\varepsilon) = \Theta(\varepsilon - \|\vec{x}_i - \vec{x}_j\|), \quad i, j = 1, \dots, N, \tag{12}$$

where  $N$  is the number of measured points  $\vec{x}_i$ ,  $\varepsilon$  is a threshold distance,  $\Theta(\cdot)$  is the Heaviside function (i.e.  $\Theta(x) = 0$ , if  $x < 0$ , and  $\Theta(x) = 1$  otherwise) and  $\|\cdot\|$  is a norm.

RPs for various values of the *message* exhibit visually the recurrences of the QHF-based PRNG (Supplementary Fig. S5 online). We used an embedding dimension  $m = 2$  and the delay  $\tau = 1$ . The threshold distance  $\varepsilon$  is set to be 10% of the mean phase space radius  $\sigma$ . It is shown that the QHF-based PRNG with different *messages* causes a rather homogeneous RP with numerous single points and some short, diagonal or vertical lines.

Because the visual impact produced by the RP is insufficient to demonstrate the quality of the QHF-based PRNG because of the ‘small-scale’ structures<sup>29</sup>, several measures of complexity which quantify the small scale structures in RPs, have been proposed<sup>34–36</sup> and are known as recurrence quantification analysis (RQA). In this paper, these measures based on the recurrence point density and the diagonal and vertical line structures are considered.

**Measures based on the recurrence density.** The simplest measure of the RQA is the recurrence rate (RR)

$$RR(\varepsilon) = \frac{1}{N^2} \sum_{i,j=1}^N R_{i,j}(\varepsilon), \tag{13}$$

which is a measure of the density of recurrence points in the RP. Furthermore, in the limit  $N \rightarrow \infty$ , RR is the probability that a state recurs to its  $\varepsilon$ -neighbourhood in phase space. For PRNGs, the ideal value would be  $RR = 0$ . But in practice, in order that the quantifier may make sense, a larger  $\varepsilon$  should be adopted to avoid the situation in which no points are found in the recurrence plot. It is shown that the value of the RR ranges from 0.00595 to 0.00662 for different *messages* (see Supplementary Fig. S6 online). It exhibits the good randomness of the QHF-based PRNG.

**Measures based on diagonal lines.** The measures are related to the histogram  $P(\varepsilon, l)$  of the diagonal line lengths  $l$ , given by

$$P(\varepsilon, l) = \sum_{i,j=1}^N (1 - R_{i-1,j-1}(\varepsilon))(1 - R_{i+l,j+l}(\varepsilon)) \prod_{k=0}^{l-1} R_{i+k,j+k}(\varepsilon). \tag{14}$$

Supplementary Fig. S7 online shows the histogram of the diagonal line lengths of the *RP* in Supplementary Fig. S5 online with the parameter *message* = 50. It is shown that the diagonal line lengths are mainly very short exhibiting the good randomness.

Processes with uncorrelated or weakly correlated behaviors cause none or very short diagonals, whereas deterministic processes cause longer diagonals and less single, isolated recurrence points. Therefore, the ratio of recurrence points that form diagonal structures (of at least length  $l_{\min}$ ) to all recurrence points

$$DET = \frac{\sum_{l=l_{\min}}^N IP(\varepsilon, l)}{\sum_{l=1}^N IP(\varepsilon, l)}, \quad (15)$$

is introduced as a measure for determinism (or predictability) of the system. The threshold  $l_{\min}$  excludes the diagonal lines which are formed by the tangential motion of the phase space trajectory.

A diagonal line of length  $l$  means that a segment of the trajectory is rather close during  $l$  time steps to another segment of the trajectory at a different time; thus these lines are related to the divergence of the trajectory segments. The average diagonal line length

$$L = \frac{\sum_{l=l_{\min}}^N l IP(\varepsilon, l)}{\sum_{l=l_{\min}}^N IP(\varepsilon, l)}, \quad (16)$$

is the average time that two segments of the trajectory are close to each other, and can be interpreted as the mean prediction time.

Another *RQA* measure considers the length  $L_{\max}$  of the longest diagonal line found in the *RP*,

$$L_{\max} = \max(\{l_i\}_{i=1}^{N_l}), \quad (17)$$

where  $N_l = \sum_{l=l_{\min}}^N P(\varepsilon, l)$  is the total number of diagonal lines. These measures are related to the exponential divergence of the phase space trajectory. The faster the trajectory segments diverge, the shorter the diagonal lines are.

The measure entropy refers to the Shannon entropy of the probability  $p(l) = P(\varepsilon, l)/N_l$  to find a diagonal line of exactly length  $l$  in the *RP*, where  $N_l = \sum_{l \geq l_{\min}} P(\varepsilon, l)$  is the total number of diagonal lines.

$$ENTR = - \sum_{l=l_{\min}}^N p(l) \ln p(l). \quad (18)$$

*ENTR* reflects the complexity of the *RP* in respect of the diagonal lines, e.g. for uncorrelated noise the value of *ENTR* is rather small, indicating its low complexity, as shown in Supplementary Fig. S8 online.

**Measures based on vertical lines.** The total number of the vertical lines of the length  $v$  in the *RP* is then given by the histogram

$$P(v) = \sum_{i,j=1}^N (1 - R_{i,j}(\varepsilon))(1 - R_{i,j+v}(\varepsilon)) \prod_{k=0}^{v-1} R_{i,j+k}(\varepsilon). \quad (19)$$

Supplementary Fig. S9 online shows the histogram of vertical line lengths of the *RP* in Supplementary Fig. S5 with the parameter *message* = 50. It is shown that the vertical line lengths are mainly very short exhibiting the good randomness.

Analogous to the definition of the determinism in equation (15), the ratio between the recurrence points forming the vertical structures and the entire set of recurrence points can be computed,

$$LAM = \frac{\sum_{v=v_{\min}}^N v P(v)}{\sum_{v=1}^N v P(v)}. \quad (20)$$

The computation of *LAM* is realized for those  $v$  that exceed a minimal length  $v_{\min}$  in order to decrease the influence of the tangential motion. *LAM* will decrease if the *RP* consists of more single recurrence points than vertical structures.

The average length of vertical structures is given by

$$TT = \frac{\sum_{v=v_{\min}}^N v^2 P(v)}{\sum_{v=v_{\min}}^N v P(v)}, \quad (21)$$

and is called trapping time. *TT* estimates the mean time that the system will abide at a specific state or how long the state will be trapped.

Finally, the maximal length of the vertical lines in the *RP*

$$V_{\max} = \max(\{v_l\}_{l=1}^{N_v}), \quad (22)$$

can be defined, analogously to the standard measure  $L_{\max}$  ( $N_v$  is the absolute number of vertical lines).



Figures 4–6 give some selected RQA measures for different values of the *message* and demonstrates the good statistical properties of the QHF-based PRNG.

**Degree of non-periodicity.** In order to study the non-periodicity in the QHF-based PRNG, the scale index analysis (SIA) is carried out which is introduced by Benitez *et al.*<sup>37</sup>. The SIA method is often used as a framework to enhance the general performance of cryptosystems in designing new chaos-based cryptosystems and PRNGs. For example, recently Akhshani *et al.* proposed a new scheme for generating good PRNGs based on quantum logistic map<sup>38</sup>. They used the SIA technique to assess the degree of non-periodicity of the chaotic sequences of the quantum map. The SIA technique is based on the continuous wavelet transform (CWT) and the wavelet multi-resolution analysis<sup>39</sup>. To study the non-periodicity of the QHF-based PRNG<sup>40</sup>, we assumed that the key sequence  $f$  generated by QHF is compactly supported and is defined over a finite time interval  $T = [a, b]$ . The CWT of  $f$  at time  $u$  and scale  $s$  is defined as<sup>39</sup>:

$$Wf(u, s) := \langle f, \psi_{u,s} \rangle = \int_{-\infty}^{+\infty} f(t) \psi_{u,s}^*(t) dt, \tag{23}$$

and it provides the frequency component (or details) of  $f$  corresponding to the scale  $s$  and time location  $t$ . Supplementary Fig. S10 online shows the time frequency decomposition of  $f$  in the time-frequency plane which provided by the wavelet transform given in equation (23).

The wavelet transform given in equation (23), provides a time frequency decomposition of  $f$  in the time-frequency plane.

The scalogram of  $f$  is defined as follows:

$$\zeta(s) := \|Wf(u, s)\| = \left( \int_{-\infty}^{+\infty} |Wf(u, s)|^2 du \right)^{\frac{1}{2}}, \tag{24}$$

where  $\zeta(s)$  is the energy of the CWT of  $f$  at scale  $s$ . The scalogram is a useful tool for studying a signal, since it allows the detection of its most representative scales or frequencies<sup>37,40</sup>. The innerscalogram of  $f$  at a scale  $s$  can be defined by:

$$\zeta^{inner}(s) := \|Wf(u, s)\|_{J(s)} = \left( \int_{c(s)}^{d(s)} |Wf(u, s)|^2 du \right)^{\frac{1}{2}}, \tag{25}$$

where  $J(s) = [c(s), d(s)] \subseteq T$  is the maximal subinterval in  $T$  for which the support of  $\psi_{u,s}$  is included in  $T$  for all  $u \in J(s)$ . As the length of  $J(s)$  depends on the scale  $s$ , the values of the inner scalogram at different scales cannot be compared. Therefore, the inner scalogram should be normalized as follows<sup>37</sup>:

$$\bar{\zeta}^{inner}(s) = \frac{\zeta^{inner}(s)}{(d(s) - c(s))^{\frac{1}{2}}}. \tag{26}$$

Supplementary Fig. S11 online shows that the normalized inner scalogram can be a valuable tool for detecting the non-periodicity of the signals, where a signal with details at every scale is non-periodic.

The scale index of  $f$  in the scale interval  $[s_0, s_1]$  can be defined by:

$$i_{scale} := \frac{\zeta(s_{min})}{\zeta(s_{max})}, \tag{27}$$

where  $s_{max}$  is the smallest scale such that  $\zeta(s) \leq \zeta(s_{max})$  for all  $s \in [s_0, s_1]$ , and  $s_{min}$  the smallest scale such that  $\zeta(s_{min}) \leq \zeta(s)$  for all  $s \in [s_{max}, s_1]$ . The scale index will be zero or close to zero for periodic sequences and close to one for highly non-periodic sequences<sup>37</sup>.

In Fig. 7, the SIA of the QHF-based key sequence is presented. It can be concluded that the best value of the scale index is up to 0.98. Thus, the key sequence in this state is highly non-periodic and it can be used for any PRNG purposes.

**The application of the QHF-based PRNG to image encryption.** The proposed image encryption algorithm consists of the following major steps:

*Step 1: Initialization step.* In this step we choose the parameters  $(n, (\alpha, \beta, \chi, \delta))$  and the *message*, then generate a random sequence  $S$  by running our proposed PRNG given by

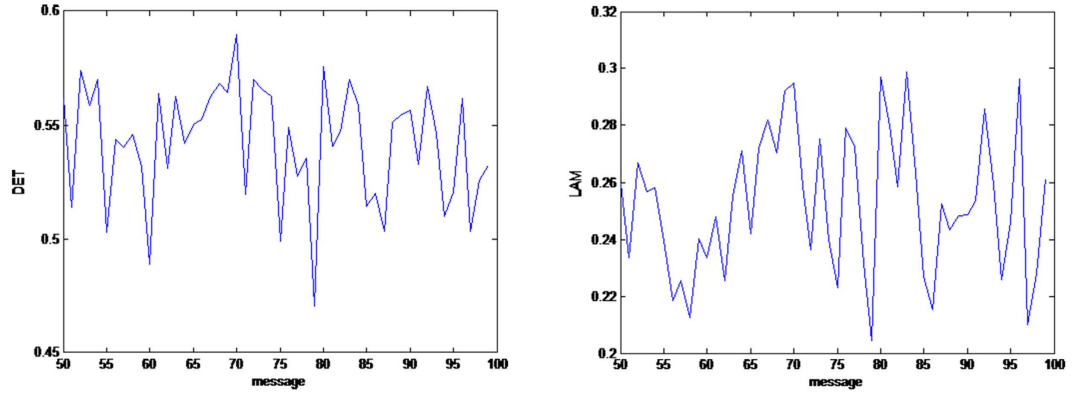
$$S = PRNG(n, (\alpha, \beta, \chi, \delta), message). \tag{28}$$

Each value of the sequence  $S$  is a floating-point number ranging from 0 to 1.

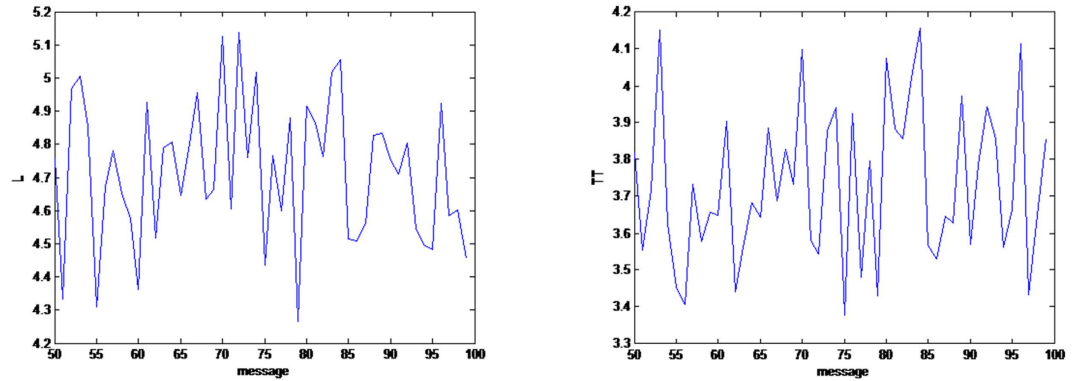
*Step 2: Encryption procedure.*

- (1) Divide the original image  $I$  with size  $M \times N$  into four parts with the same size and determine the encryption order of four image blocks according to the size of the first four values of the sequence  $S$ . For example,  $I_1 \rightarrow I_2 \rightarrow I_3 \rightarrow I_4$  if  $P(1, 1) > P(1, 2) > P(1, 3) > P(1, 4)$ . Here we take Buddha image as an example





**Figure 4.** Selected RQA measures  $DET$ ,  $L_{max} \cdot DET$  and  $LAM$  change with different *messages* are shown in (a,b) respectively. (see text in the section entitled Results).



**Figure 5.** Selected RQA measures  $L$  and  $TT$ .  $L$  and  $TT$  change with different *messages* are shown in (a,b) respectively. (see text in the section entitled Results).

(see Supplementary Fig. S12 online).

(2) The encryption process is a two round one (Supplementary Fig. S13 online) i.e.,

$$\begin{aligned}
 EI_1 &= \text{encrypt}(I_1, I_4, S) \rightarrow EI_2 = \text{encrypt}(I_2, EI_1, S) \\
 \rightarrow EI_3 &= \text{encrypt}(I_3, EI_2, S) \rightarrow EI_4 = \text{encrypt}(I_4, EI_3, S) \\
 \rightarrow EI'_1 &= \text{encrypt}(EI_1, EI_4, S) \rightarrow EI'_2 = \text{encrypt}(EI_2, EI'_1, S) \\
 \rightarrow EI'_3 &= \text{encrypt}(EI_3, EI'_2, S) \rightarrow EI'_4 = \text{encrypt}(EI_4, EI'_3, S)
 \end{aligned} \tag{29}$$

Then we describe the encrypt function  $\text{encrypt}(*, *, *)$  by taking  $I_2$  as an example.

1. Sum the gray values of  $EI_1$  to produce a binary sequence  $V$ , use  $V$  as the *message* to generate a new random sequence  $S_1$  as follows.

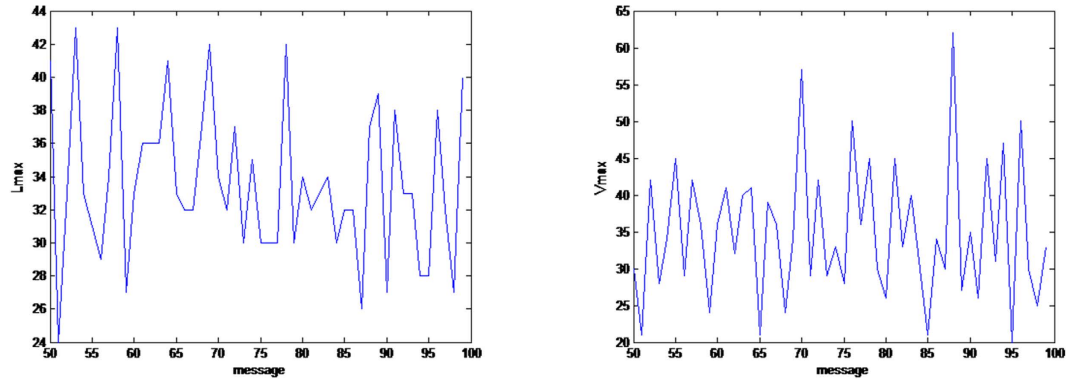
$$\text{sum} = \sum_{i=1, j=1}^{i=M/2, j=N/2} \text{pixel}_{i,j} \tag{30}$$

where  $\text{pixel}_{i,j}$  represents the gray value of the pixel in the  $i^{\text{th}}$  row and  $j^{\text{th}}$  column of  $EI_1$ .

$$S_1 = \text{PRNG}(n, (\alpha, \beta, \chi, \delta), \text{dec2bin}(\text{sum})), \tag{31}$$

where  $\text{dec2bin}$  is a function to convert the decimal to binary sequence.

2. Do the tensor product of  $S$  and  $S_1$  to obtain a new sequence and transform it into a two-dimensional matrix of size  $(M/2) \times (N/2)$ .



**Figure 6.** Selected RQA measures  $L_{max}$  and  $V_{max}$ .  $L_{max}$  and  $V_{max}$  change with different *messages* are shown in (a,b) respectively. (see text in the section entitled Results).

$$P = \text{reshape}\left(S \otimes S_1, \frac{M}{2}, \frac{N}{2}\right), \tag{32}$$

where  $\otimes$  is tensor product operator.

3. Multiply all values in the  $P$  matrix by  $10^8$  modulo 256 to form a new matrix  $P'$ .

$$P = \text{mod}(\text{floor}(p \cdot 10^8), 256), \tag{33}$$

4. Make  $P'$  exclusive-OR (XOR) with  $I_2$  and get

$$\text{temp\_}I_2 = I_2 \oplus P'. \tag{34}$$

5. Sort the elements in  $P'$  in ascending order, and obtain the index vector.

$$[\text{sorted}P, \text{index}] = \text{sort}(P), \tag{35}$$

where  $\text{sorted}P$  is the sorted matrix and  $\text{index}$  is the index vector, i.e.,  $\text{index} = \{h_1, h_2, \dots, h_{\frac{MN}{4}}\}$ .

6. Shuffle the elements in  $\text{temp\_}I_2$  and get encrypted image block  $EI_2$  for the first round.

$$EI_2(h_i) = \text{temp\_}I_2(i), \quad i = 1, 2, \dots, \frac{MN}{4}. \tag{36}$$

At last we can get the final encrypted image by executing the encrypt function as the order shown in *Step 2*.

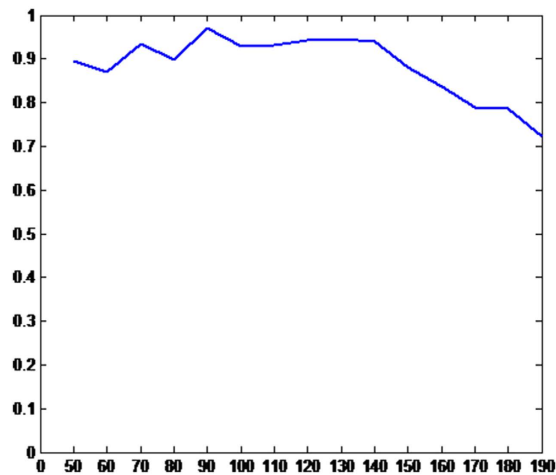
*Step 3: Decryption procedure.* In the decryption process, we use the secret key to determine the decryption order, shown as follows.

$$\begin{aligned} EI_4 &= EI_3' \otimes EI_4' \rightarrow EI_3 = EI_2' \otimes EI_3' \rightarrow EI_2 = EI_1' \otimes EI_2' \rightarrow EI_1 = EI_4 \otimes EI_1' \\ &\rightarrow I_4 = EI_3 \otimes EI_4 \rightarrow I_3 = EI_2 \otimes EI_3 \rightarrow I_2 = EI_1 \otimes EI_2 \rightarrow I_1 = I_4 \otimes EI_1 \rightarrow \text{end} \end{aligned} \tag{37}$$

Here,  $\otimes$  represents the decryption operation and it is the reverse of the encryption process.

**Experimental simulations.** Experiments are performed on a laptop with Intel(R) Core(TM) i3-2370M CPU 2.40 GHz 2 GB RAM running on Windows 7 professional equipped with the MATLAB R2012a environment and the Python 2.7.8. To test our encryption method for security and robustness, we choose a group of images with size  $512 \times 512$ , which were taken by Yu-Guang Yang and list them and their corresponding encrypted images and decrypted images in Supplementary Figs S12–S18.

**Key space analysis.** A desirable encryption scheme should have a sufficiently large key space to resist brute-force attacks. Here the encryption key can be represented by  $(n, (\alpha, \beta, \chi, \delta), \text{message})$ , where  $n$  denotes the number of nodes in a circle.  $\alpha, \beta, \chi, \delta$  are complex numbers and also control parameters of the coin state satisfying the normalization constraint:  $|\alpha|^2 + |\beta|^2 + |\chi|^2 + |\delta|^2 = 1$ . *Message* is a binary string as an input of the proposed QHF, which can have an infinite length theoretically. Although there is an infinite key space theoretically, because of finite precision of digital computers, the key space actually turns out to be finite. Considering that the calculation precision is  $10^{-14}$ , the size of key space for initial conditions and control parameters would be



**Figure 7.** The scale index of the QHF-based key sequence for different messages. It can be concluded that, the value of the scale index is larger than 0.7 and the best value of the scale index is up to 0.98. Thus, the key sequence in this state is highly non-periodic and it can be used for any PRNG purposes. (see text in the section entitled Results).

roughly  $2^{325}$ , which is large enough for any encryption purposes and is also large enough to resist all kinds of brute-force attacks.

**Histogram analysis.** Histogram is a very important security measure for evaluating the security of an image encryption algorithm. The histograms of seven images and the corresponding cipher images are shown in Supplementary Figs S12–S18 online. It is shown that the histograms of the cipher images are nearly uniform and significantly different from the ones of the original images. Therefore they provide no clue for attackers in a statistical analysis attack.

**Correlation analysis.** A desirable encryption scheme should generate the cipher image with rather low correlation between adjacent pixels. By randomly selecting 10000 pairs (in horizontal, vertical and diagonal directions respectively) of adjacent pixels from the original image and the cipher image, respectively, we test the correlation between adjacent pixels, and draw the correlation distribution of adjacent pixels in the Buddha image and its cipher image in Supplementary Fig. S21 online, respectively. It is shown that the original image has strong correlation, but the cipher image is quite random. The encryption scheme improves the security of the test images greatly.

We also calculate the correlation coefficient  $r_{xy}$  of adjacent pixels of the original image and the cipher image given by

$$r_{xy} = \frac{E((x - E(x))(y - E(y)))}{\sqrt{D(x)D(y)}}, \quad (38)$$

where  $E(x)$  and  $D(x)$  are the expectation and variance of variable  $x$ , respectively.

We compute the correlation coefficients of the seven images and their cipher images. From Supplementary Table S2 online, we can see that the average correlation coefficients of the encrypted images are 0.0010, 0.0012, 0.0023 and they are very close to zero. The result demonstrates that our scheme is effective.

**Comparison with other image encryption techniques.** Experimental results of the proposed image encryption scheme were compared with six typical image encryption techniques, i.e., chaos-based image encryption<sup>41</sup>, optics-based image encryption<sup>42</sup>, hash-based image encryption<sup>43</sup>, quantum image encryption<sup>44</sup>, the QW-based image encryption<sup>14</sup>, and the image encryption based on quantum logistic map<sup>45</sup>, respectively. From Supplementary Tables S3–S8 online, we can see that the images encrypted by our algorithm have lower correlation compared with the six image encryption schemes<sup>14,41–45</sup>. In addition, we also compared our algorithm with two recently published schemes<sup>46,47</sup> and see Supplementary Tables S9 and S10 online for details.

**Differential attack analysis.** In general, two common performance measures are used to test the influence of a little change in the original image on the cipher image, i.e., the *number of pixels change rate* (NPCR) and the *unified average changing intensity* (UACI). NPCR is expressed by

$$NPCR = \frac{\sum_{i=1}^m \sum_{j=1}^n D(i, j)}{m \times n} \times 100\%, \quad (39)$$

where

$$D(i, j) = \begin{cases} 0 & c_1(i, j) = c_2(i, j) \\ 1 & \text{otherwise} \end{cases} \quad (40)$$

and  $c_1$  and  $c_2$  are two cipher images with size  $m \times n$ .

$UACI$  is defined by

$$UACI = \frac{1}{m \times n} \left| \frac{\sum_{i=1}^m \sum_{j=1}^n (c_1(i, j) - c_2(i, j))}{255} \right| \times 100\%. \quad (41)$$

In our tests, we just put the gray value of grid (100,100) minus one in the seven images in Supplementary Figs S12–S18 with size  $512 \times 512$ , respectively. From Supplementary Table S11 online, we can see that  $NPCR$  is 99.61% and  $UACI$  is 33.46% in our algorithm which implies that it is highly sensitive to the original image and is robust against differential attacks. In addition, we also compared our proposal with other image encryption schemes<sup>41,43,45,46,47</sup> and see Supplementary Tables S12–S17 online for details. From these tables, it is indicated that the proposed algorithm can resist the differential attack. The secret key in our algorithm is related to the image self. So it makes a good performance in this field.

**Key sensitivity analysis.** First, we encrypted Buddha image with the key  $n = 1$ ,  $[\alpha, \beta, \chi, \delta] = [0, 0, 0, 1]$ , message = [1, 0, 1, 0, ..., 1, 0, 1, 0...], and obtained the cipher image in Supplementary Fig. S22(a). Then we encrypted Buddha image by making a little change of the message and got the cipher image in Supplementary Fig. S22(b). We also draw the differential image of Supplementary Fig. S22(a) and S22(b) in Supplementary Fig. S22(c). By calculation, we got that the difference ( $NPCR$ ) between Supplementary Fig. S22(a) and Supplementary Fig. S22(b) is 99.59% (see Supplementary Tables S18 online), which implies the encryption process is quite sensitive to the encryption key.

Second, we encrypted Buddha image by the key  $n = 1$ ,  $[\alpha, \beta, \chi, \delta] = [0, 0, 0, 1]$ , message = [1, 0, 0, 1, ..., 1, 0, 1, 1...], and then decrypted the resulting cipher image with the correct key (see Supplementary Fig. S22(d)) and the wrong key with a little change of the message = [1, 0, 0, 1, ..., 1, 1, 1, 1...], (see Supplementary Fig. S22(e)), respectively. We calculated out the difference ( $NPCR$ ) between Supplementary Fig. S22(d) and Fig. S22(e) is 99.61% (see Supplementary Tables S18 online). Therefore, the decryption process is also highly sensitive to the decryption key.

In the same way, we also calculated the  $NPCR$  of other test images from Supplementary Figs S12–S18, and listed the results in Supplementary Table S18 online. It is found that the results are approximately 99.60%, which shows that the proposed algorithm is of good key sensitivity to images.

**Information entropy analysis.** The information entropy is often used to measure the randomness of the cipher images. The entropy  $H(x)$  of a message source  $m$  is given by

$$H(X) = - \sum_{i=0}^{L-1} p(x_i) \log_2 p(x_i), \quad (42)$$

where  $p(x_i)$  represents the probability of the occurrence of symbol  $x_i$ .

From Supplementary Table S19 online, we can see the average information entropy of our cipher images is very close to the theoretical value 8. We also compared our algorithm with the algorithm based on quantum logistic map<sup>45</sup> in terms of information entropy (Supplementary Table S20 online). This implies that the information leakage in the encryption process is negligible and the proposed algorithm is stable and secure against information entropy attack.

**Randomness test analysis.** Next, we used NIST SP800-22 and TestU01<sup>48</sup> tools to test the randomness of the cipher images. Each test produces a  $P$ -value in [0, 1]. If the  $P$ -value is higher than a preset threshold  $\alpha$ , it means that the cipher image passes the test. In our tests, we set  $\alpha = 0.01$  and Buddha's cipher image with size  $1024 \times 1024$ .  $\alpha = 0.01$  implies that the cipher image can be inferred to be random with 99% probability if it passes the test. From Supplementary Table S21 online, we can judge that our proposed algorithm passes the NIST SP800-22 tests.

However, when we applied the most stringent test by TestU01<sup>48</sup> on the cipher images, surprisingly, the TestU01 test cannot be done successfully. Maybe the coin flipping operators were not chosen properly. Therefore, the future work will focus on the improvement of our proposal.

**Speed performance analysis.** Speed is an important factor for evaluating the performance of an image encryption algorithm. For the proposed image encryption algorithm, we measured the time cost in the operating environment: Windows 7, Matlab R2012a, Intel(R) Core(TM) i3-2370M CPU 2.40 GHz 4 GB RAM. In our algorithm, the time is mainly spent on shuffling pixels, we can figure out that the time complexity of our algorithm is  $T(n) = O(n^2)$  and the average time cost for encrypting images of size  $512 \times 512$  is 0.015 seconds or so. Therefore, our algorithm is fast enough for practical applications.

## Discussion

The main contribution of the work is to construct a hash function used for the privacy amplification process of QKD systems with higher security by means of the physical principles of quantum mechanics. As a byproduct,

QHF can also be used for pseudo-random number generation due to its inherent chaotic dynamics and further we propose a novel QHF-based image encryption algorithm.

The QHF is in fact constructed in the quantum context. Because the practical quantum computation device is unavailable, we have to do the simulations by MATLAB software on a classical computer in order to demonstrate the performance of the constructed QHF. However, in fact the properties of quantum parallel computation cannot be simulated precisely on a classical computer. So both randomness and speed of the proposed scheme seem not acceptable for cryptographic purpose. Fortunately, with the rapid development of the field of quantum computation, maybe the practical quantum computer is possible in the future. When the QHF operates on a practical quantum computer, the computation speed will be increased exponentially because of the properties of quantum parallel computation.

## Methods

**The construction of a QHF by modifying the one-dimensional two-particle discrete-time QW on a circle.** A one-dimensional two-particle discrete-time QW on a circle describes the QW of two walkers whose motions are restricted to the circle. The operators  $\hat{S}_1$  and  $\hat{S}_2$  of two-particle QW on circles becomes

$$\hat{S}_1 = \begin{cases} |2, 0\rangle\langle 1, 0| + |n, 1\rangle\langle 1, 1|, & \text{when } x = 1; \\ |1, 0\rangle\langle n, 0| + |n-1, 1\rangle\langle n, 1|, & \text{when } x = n; \\ |x+1, 0\rangle\langle x, 0| + |x-1, 1\rangle\langle x, 1|, & \text{when } x \neq 1, n. \end{cases} \quad (43)$$

Here  $\hat{S}_2$  is similar to  $\hat{S}_1$ . The total conditional sift operator  $\hat{S}$  is denoted as  $\hat{S} = \hat{S}_1 \otimes \hat{S}_2$ .

When the  $i$ th bit of the message is 0(1), the  $i$ th step of the walk executes with the interaction  $\hat{C}_0(\hat{C}_1)$ . For example, if the message,  $m$  is '0100110', then the final state evolves

$$|\psi\rangle_7 = \hat{U}_0 \hat{U}_1 \hat{U}_1 \hat{U}_0 \hat{U}_0 \hat{U}_1 \hat{U}_0 |\psi\rangle_0, \quad (44)$$

where  $\hat{U}_0 = \hat{S}(\hat{I} \otimes \hat{C}_0)$  and  $\hat{U}_1 = \hat{S}(\hat{I} \otimes \hat{C}_1)$ . The initial state of the total quantum system  $|\psi\rangle_0$  is given by

$$|\psi\rangle_0 = |x, y\rangle \otimes |v_1, v_2\rangle. \quad (45)$$

Here

$$|v_1, v_2\rangle = (\alpha|00\rangle + \beta|01\rangle + \chi|10\rangle + \delta|11\rangle), \quad (46)$$

where  $|\alpha|^2 + |\beta|^2 + |\chi|^2 + |\delta|^2 = 1$ .

## References

- Bennett, C. H., Brassard, G., Crépeau, C. & Maurer, U. M. Generalized privacy amplification. *IEEE Trans. Inf. Theory* **41**, 1915–1923 (1995).
- Bennett, C. H. & Brassard, G. Quantum cryptography: public key distribution and coin tossing. In: *Proceedings of IEEE International Conference on Computers, Systems, and Signal Processing, IEEE, New York*, pp. 175–179 (1984).
- Ekert, A. K. Quantum cryptography based on Bell's theorem. *Phys. Rev. Lett.* **67**, 661–663 (1991).
- Tamaki, K., Koashi, M. & Imoto, N. Unconditionally secure key distribution based on two nonorthogonal states. *Phys. Rev. Lett.* **90**, 167904 (2003).
- Elías, S. & Andraca, V. Quantum walks: a comprehensive review. *Quantum Inf. Process.* **11**, 1015–1106 (2012).
- Ambainis, A. Quantum walk algorithm for element distinctness. *SIAM J. Comput.* **37**, 210–239 (2007).
- Magniez, F., Santha, M. & Szegedy, M. Quantum algorithms for the triangle problem. *SIAM J. Comput.* **37**, 413–424 (2007).
- Li, Q., He, Y. & Jiang, J.-P. A hybrid classical-quantum clustering algorithm based on quantum walks. *Quantum Inf. Process.* **10**, 13–26 (2011).
- Babatunde, A. M., Cresser, J. & Twamley, J. Using a biased quantum random walk as a quantum lumped element router. *Phys. Rev. A* **90**, 012339 (2014).
- Tamascelli, D. & Zanetti, L. A quantum-walk-inspired adiabatic algorithm for solving graph isomorphism problems. *J. Phys. A-Math. Theor.* **47**, 325302 (2014).
- Childs, A. M. & Ge, Y. M. Spatial search by continuous-time quantum walks on crystal lattices. *Phys. Rev. A* **89**, 052337(2014).
- Izaac, J. A. & Wang, J. B. pyCTQW: A continuous-time quantum walk simulator on distributed memory computers. *Comput. Phys. Commun.* **186**, 81–92 (2015).
- Zhan, X., Qin, H., Bian, Z. H., Li, J. & Xue, P. Perfect state transfer and efficient quantum routing: A discrete-time quantum-walk approach. *Phys. Rev. A* **90**, 012331 (2014).
- Yang, Y. G., Pan, Q. X., Sun, S. J. & Xu, P. Novel image encryption based on quantum walks. *Sci. Rep.* **5**, 7784 (2015).
- Li, D., Zhang, J., Guo, F.-Z., Huang, W. & Wen, Q.-Y. Discrete-time interacting quantum walks and quantum Hash schemes. *Quantum Inf. Process.* **12**, 1501–1513 (2013).
- Shenvi, N., Kempe, J. & Whaley, K. B. Quantum random-walk search algorithm. *Phys. Rev. A* **67**, 052307 (2003).
- Štefaňák, M., Barnett, S. M., Kollár, B., Kiss, T. & Jex, I. Directional correlations in quantum walks with two particles. *New J. Phys.* **13**, 033029 (2011).
- Zhang, J., Wang, X. & Zhang, W. Chaotic keyed hash function based on feed forward-feedback nonlinear digital filter. *Phys. Lett. A* **362**, 439–448 (2007).
- Rivest, R. The MD5 message-digest algorithm. *RFC 1321* <http://www.fourmilab.ch/hotbits> (1992) (29/6/2015).
- Nielsen, M. & Chuang, I. *Quantum Computation and Quantum Information*. Cambridge University Press, New York, 2000.
- Akhavan, A., Samsudin, A. & Akhshani, A. A novel parallel hash function based on 3D chaotic map. *EURASIP J. Adv. Signal Process.* **2013**, 126 (2013).
- Akhshani, A. et al. Hash function based on hierarchy of 2D piecewise nonlinear chaotic maps. *Chaos, Solitons and Fractals* **42**, 2405–2412 (2009).
- Wei, G., Wang, X. M., He, D. K. & Cao, Y. Cryptanalysis on a parallel keyed hash function based on chaotic maps. *Phys. Lett. A* **373**, 3201–3206 (2009).

24. Wang, X.-Y. & Zhao, J.-F. Cryptanalysis on a parallel keyed hash function based on chaotic neural network. *Neurocomputing* **73**, 3224–3228 (2010).
25. López-Ruiz, R., Mancini, H. L. & Calbet, X. A statistical measure of complexity. *Phys. Lett. A* **209**, 321–326 (1995).
26. Lambert, P. W., Martin, M. T., Piastino, A. & Rosso, O. A. Intensive entropy non-triviality measure. *Physica A* **334**, 119–131 (2004).
27. Rosso, O. A., Larrondo, H. A., Martin, M. T., Plastino, A. & Fuentes, M. A. Distinguishing noise from chaos. *Phys. Rev. Lett.* **99**, 154102 (2007).
28. Eckmann, J. P., Oliffson Kamphorst, S. & Ruelle, D. Recurrence plots of dynamical systems. *Europhys. Lett.* **4**, 973–977 (1987).
29. Marwan, N., Romano, M. C., Thiel, M. & Kurths, J. Recurrence plots for the analysis of complex systems. *Phys. Rep.* **438**, 237–329 (2007).
30. Shiner, J. S., Davison, M. & Landsberg, P. T. Simple measure for complexity. *Phys. Rev. E* **59**, 1459–1464 (1999).
31. Martin, M. T., Plastino, A. & Rosso, O. A. Statistical complexity and disequilibrium. *Phys. Lett. A* **311**, 126–132 (2003).
32. Larrondo, H. A., González, C. M., Martin, M. T., Plastino, A. & Rosso, O. A. Intensive statistical complexity measure of pseudorandom number generators. *Physica A* **356**, 133–138 (2005).
33. Bandt, C. & Pompe, B. Permutation Entropy: A natural complexity measure for time series. *Phys. Rev. Lett.* **88**, 174102 (2002).
34. Marwan, N., Wessel, N., Meyerfeldt, U., Schirdewan, A. & Kurths, J. Recurrence plot based measures of complexity and its application to heart rate variability data. *Phys. Rev. E* **66**, 026702 (2002).
35. Zbilut, J. P. & Webber, C. L. Embeddings and delays as derived from quantification of recurrence plots. *Phys. Lett. A* **171**, 199–203 (1992).
36. Webber, C. L. & Zbilut, J. P. Dynamical assessment of physiological systems and states using recurrence plot strategies. *J. Appl. Physiol.* **76**, 965–973 (1994).
37. Benítez, R., Bolós, V. J. & Ramírez, M. E. A wavelet-based tool for studying non-periodicity. *Comput. Math. Appl.* **60**, 634–641 (2010).
38. Akhshani, A., Akhavan, A., Mobaraki, A., Lim, S.-C. & Hassan, Z. Pseudo random number generator based on quantum chaotic map. *Commun. Nonlinear Sci. Numer. Simulat.* **19**, 101–111 (2014).
39. Mallat, S. *A wavelet tour of signal processing*. Academic Press London (1999).
40. Chandre, C., Wiggins, S. & Üzer, T. Time-frequency analysis of chaotic systems. *Physica D* **181**, 171–196 (2003).
41. Gao, T. & Chen, Z. A new image encryption algorithm based on hyper-chaos. *Phys. Lett. A* **372**, 394–400 (2008).
42. Refregier, P. & Javidi, B. Optical image encryption based on input plane and Fourier plane random encoding. *Opt. Lett.* **20**, 767–769 (1995).
43. Cheddad, A., Condell, J., Curran, K. & McKeivitt, P. A hash-based image encryption algorithm. *Opt. Commun.* **283**, 879–893 (2010).
44. Yang, Y.-G., Xia, J., Jia, X. & Zhang, H. Novel image encryption/decryption based on quantum Fourier transform and double phase encoding. *Quantum Inf. Process.* **12**, 3477–3493 (2013).
45. Akhshani, A., Akhavan, A., Lim, S.-C. & Hassan, Z. An image encryption scheme based on quantum logistic map. *Commun. Nonlinear Sci. Numer. Simulat.* **17**, 4653–4661 (2012).
46. Huang, X. L. & Ye, G. D. An image encryption algorithm based on hyper-chaos and DNA sequence. *Multimed. Tools Appl.* **72**, 57–70 (2014).
47. Shahram, E. G. & Mohammad, E. Chaotic image encryption system using phase-magnitude transformation and pixel substitution. *Telecommun. Syst.* **52**, 525–537 (2013).
48. L'Ecuyer, P. & Simard, R. J. TestU01: A C library for empirical testing of random number generators. *ACM Trans. Math. Soft.* **33**, 22 (2007).

## Acknowledgements

This work was supported by the National Natural Science Foundation of China (Grant No.61572053); The Importation and Development of High-Caliber Talents Project of Beijing Municipal Institutions (No. CIT&TCD201304039); The Scientific Research Common Program of Beijing Municipal Commission of Education (No.KM201510005016); The Basic Research Foundation of Beijing University of Technology (Nos.X4 007999201501,007000514315501); Jing-Hua Talents Project of Beijing University of Technology(2014-JH-L06); Beijing Natural Science Foundation (Grant No.4152038).

## Author Contributions

Y.-G.Y. proposed the theoretical method and wrote the main manuscript text. P.X. made the numerical simulations. R.Y., Y.-H.Z. and W.-M.S. reviewed the manuscript.

## Additional Information

**Supplementary information** accompanies this paper at <http://www.nature.com/srep>

**Competing financial interests:** The authors declare no competing financial interests.

**How to cite this article:** Yang, Y.-G. *et al.* Quantum Hash function and its application to privacy amplification in quantum key distribution, pseudo-random number generation and image encryption. *Sci. Rep.* **6**, 19788; doi: 10.1038/srep19788 (2016).



This work is licensed under a Creative Commons Attribution 4.0 International License. The images or other third party material in this article are included in the article's Creative Commons license, unless indicated otherwise in the credit line; if the material is not included under the Creative Commons license, users will need to obtain permission from the license holder to reproduce the material. To view a copy of this license, visit <http://creativecommons.org/licenses/by/4.0/>