POLITECNICO DI TORINO
Repository ISTITUZIONALE

A Quantum Adaptation for the Morra Game and some of its Variants

Original
A Quantum Adaptation for the Morra Game and some of its Variants / Marceddu, ANTONIO COSTANTINO; Montrucchio, Bartolomeo. - In: IEEE TRANSACTIONS ON GAMES. - ISSN 2475-1510. - ELETTRONICO. - (2023), pp. 1-9. [10.1109/TG.2023.3251663]

Availability:
This version is available at: 11583/2976608 since: 2023-03-06T11:36:03Z

Publisher:
IEEE

Published
DOI:10.1109/TG.2023.3251663

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright
IEEE postprint/Author's Accepted Manuscript

©2023 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collecting works, for resale or lists, or reuse of any copyrighted component of this work in other works.

(Article begins on next page)

28 March 2024

This article has been accepted for publication in IEEE Transactions on Games. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TG.2023.3251663

A Quantum Adaptation for the Morra Game and some of its Variants

Antonio Costantino Marceddu, *Graduate Student Member, IEEE*, Bartolomeo Montrucchio, *Senior Member, IEEE*

***Abstract*—The Morra game is quite old. Back in time, traces of it can be found in ancient Egypt, ancient Rome, and even China. It involves two players who, for a limited number of turns, must try to suppose the sum of the number personally chosen with the number chosen by the opponent. The rules are simple, but it is rather difficult to play at a high level as there are multiple cognitive, motor, and perceptual processes involved.**

The goal of this paper is to illustrate the process of implementing a quantum random player for the Morra game and some of its variants. This can be done by using a quantum number generator circuit to generate two numbers and a quantum adder to obtain the supposed sum. The advantage of this proposal is that, unlike the implementations of the Morra game on classical computers, which only allow the generation of pseudo-random numbers, true randomness can be obtained through quantum computing.

In addition to the description of the entire algorithms, the source code of the implementations is provided to give everyone the freedom to easily test both the quantum implementation of the Morra game and the variants discussed in the paper.

***Index Terms*—Active perception, Cognitive processes, Games, Neuropsychology, Programming, Quantum computing**

I. Introduction

MORRA is a game with pretty simple rules. It should not be confused with the rock-paper-scissors, which is also known as Chinese Morra due to the similarities in the game mode. Morra requires that two players, at the same time, use one of their hands to mimic a number between 1 and 5 and shout another number between 2 and 10. Keeping the fist closed is equivalent to mimicking the one with a finger, while the ten is also called “Morra” or “all”. The number shouted by each player corresponds to the presumed sum of two numbers mimicked by the individual players. If one of the players supposes it, he will score a point, while if both players suppose it, no points will be awarded. The number of points to guarantee victory varies: for example, in the Sardinian version, also called “sa Murra”, they are usually 16.

The oldest traces of the existence of the game were found in the iconographies present in the tombs of Beni Hasan, Egypt [1] [2]. They were dated around 2000 BC. Later traces of the game were found in the paintings of Greek vases, most notably one depicting Helen of Troy and Paris [3], and in documents written during the Roman Empire, which was referred to using the terms “micatio” or “micare digitis” [4]. Notably, even Cicero, in the third book of his “De officiis”, refers to it in two distinct sections: 77 and 89 [5]. Due to its large extension, the Roman Empire could have favored the spread of the game itself. In China, it was known as “huaquan”, “caiquan” or “muzhan” [6]. Even today, the game continues to be called in these ways [7] [8] [9]. Nowadays, Morra is played in many countries around the world, including Italy, Spain, France, Greece, China, and Mongolia.

Despite its simple functionality, Morra is rather difficult to play [10] [11]. It requires the cooperation of multiple cognitive, motor, and perceptual processes. A good Morra player tries to:

- Learn the opponent's playstyle such that playing patterns are recognized and can be exploited. This requires careful memorization of the numbers previously said by the player himself and his adversary.
- Be unpredictable so that, in turn, the opponent avoids the recognition of game patterns to be exploited to his advantage.

Taking into account that, on a professional level, more than one round per second is played, becoming a good Morra player requires a lot of training.

Over time, several adaptations of the game have been made for Personal Computers (PC) and smartphones. They allow you to play both against Artificial Intelligence (AI) and with other players online [12]. Several robots that can play against humans have also been developed. One of them, called Gavina2121, recently challenged human opponents in Bitti, Sardinia, taking a resounding defeat [13]. However, its creators have obtained important data from the challenge, which will serve to improve the robot itself.

Quantum computers are currently of strong multidisciplinary interest as they can be faster than real computers for handling highly complex problems [14]. Another interesting feature is that, unlike classical computers, which can only generate pseudo-random numbers, they allow the generation of true random numbers [15]. In the case of Morra, this could allow for a truly unpredictable player and makes it impossible, even for an experienced player, to make any prediction of possible number generation patterns. Implementing a quantum random player for the Morra game requires knowledge of what it should do in the first place. It can therefore be assumed that it must return:

- One random number between 1 and 5, corresponding

Antonio Costantino Marceddu is with the Department of Control and Computer Engineering, Politecnico di Torino, Torino, TO 10129 Italy (e-mail: antonio.marceddu@polito.it).

Bartolomeo Montrucchio is with the Department of Control and Computer Engineering, Politecnico di Torino, Torino, TO, 10129 Italy (e-mail: bartolomeo.montrucchio@polito.it).

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Games. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TG.2023.3251663

to the number mimicked with the hand.
- One number equal to the sum of the previous number with another random number between 1 and 5, corresponding to the shouted number.

To achieve this purpose, it is necessary to create two circuits:
- One for generating two random numbers between 1 and 5.
- One for adding up the two previously generated numbers.

Based on what was previously said, this paper intends to propose a novel quantum adaptation of the Morra game and some of its variants. It is subdivided as follows:
- Section II will describe the state of the art related to quantum games and adders. The latter will be used for the implementation of Quantum Morra. In the same section, a brief introduction to quantum computing will be provided to improve the understanding of the entire work.
- Section III will discuss the implementation of the Quantum Morra.
- Sections IV and V will discuss the implementation of some variants of the Morra game.
- Finally, Section VI will draw some conclusions and future improvements.

II. Related Work

A. Quantum Games

The terminology "quantum games" is often linked with quantum game theory, which extends classical game theory toward the quantum realm. According to [16], quantum games are "*computer games where the rules of the game are based on quantum principles, such as superposition, entanglement, and the collapse of the wave function*". This concept of quantum games has been used multiple times, although it is quite limited as it concentrates solely on the educational aspect of the same. More recently, in [17] a three-dimensional characterization for the distinction of a quantum game compared to other types of games was also discussed. These dimensions concern:

- *Quantum physics*, which concerns the use of notions of quantum physics in mechanics, graphics, or other parts of games.
- *Quantum technologies*, which concern the impact of using quantum computers or software on both gameplay and game development.
- *Scientific purposes*, which aims to test the capability and limitation of quantum computers.

So, based on this characterization, quantum games can be defined as "*games that reference the theory of quantum physics, quantum technologies or quantum computing through perceivable means, connect to quantum physics through a scientific purpose or use quantum technologies*" [17]. This definition is more comprehensive and appropriate to the continual changes brought forth by this new field.

Many quantum games have been released so far. Some of them will be presented in the next lines. Cat/Box/Scissors was the first quantum game built to run on a quantum computer [18]. It is a Rock/Paper/Scissors variant in which there is a referee who decides the winner and four potential opponents to play with, although it is possible to play against one at a time. Quantum Battleship was instead the first turn-based multiplayer game made to run on a quantum computer [19]. It is a simplified version of Battleship where the first player places a ship and the second player has to try to sink it. The former wins if the ship is not sunk, while the latter wins if he manages to sink it. *Quantum Chess* is the adaptation of the classical game for quantum computers. Two different versions were proposed in [20] and [21]. The former focuses on teaching quantum mechanics properties, while the latter is more recent and focuses on being an environment to give players the ability to interact with quantum properties while playing. *Diner's Dilemma* is a classical problem related to game theory and economics. In the quantum world, it has been faced and solved in the 4-player variant [22] by implementing and testing a circuit built using IBM® Quantum Simulator. It has also been shown that players can reach both the Nash equilibrium point and the Pareto optimum point. A curious game about a unicorn attempting to fly over a castle, called *Flying Unicorn*, is presented in [23] with the aim of exploring the properties of superposition, qubit measurement, and uncertainty. Both the classical and quantum implementations of the game are provided in the paper. In [24] and [25] two implementations of *Quantum Go* have been presented. The first aimed to make quantum mechanics accessible to all, while the second aimed to become a test platform for new algorithms for AI. *Quantum Minesweeper*, differently from what can be thought of, is a classical computer game [16]. Its creators came up with it as a way of teaching basic quantum mechanics simply by playing a variation of the Minesweeper game. The *Monty Hall problem* is a probability puzzle proposed in 1975. Several quantum versions of it have been proposed over time since, in addition to being of scientific interest, its application could have a potential interest in the quantum information area [26] [27] [28]. A game similar to *Texas Hold 'em*, a variant of the game of Poker, was presented in [29] with the aim of teaching the phenomena underlying quantum computing. Also in this case, a proof-of-concept is provided with the paper. In [30], some quantum circuits were presented to solve the simplest *Sudoku* grid, 4x4 in size, through the use of the concept of duality computing and a probabilistic approach. An example code for detecting the location of new clues is also provided in the paper. Two slightly different versions of Quantum Tetris have recently been released for educational purposes [31] [32]. The former focuses mainly on teaching general notions of quantum theory, while the latter focuses on visualizing the possible effects of quantum noise on a normal run of the game. The last game that will be mentioned is *Quantum Tic Tac Toe*. In [33] it was developed as a didactic metaphor of nonlocality and quantum physics, while in [34] the goal was to create a minimal quantization of the classical game.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Games. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TG.2023.3251663



Fig. 1. Icons showing the quantum gates and the operations described as shown in the IBM Quantum Composer [39] [40]. From left to right, top to bottom: NOT gate, CNOT gate, Toffoli gate, Hadamard gate, SWAP gate, RX gate, RY gate, RZ gate, barrier operation and measurement operation.

B. Brief Introduction to Quantum Computing

In quantum computing, the *qubit* (an acronym for *quantum binary digit*) is the smallest unit of information that can be dealt with. Each obtainable state can be written as the superposition of two orthonormal basis states, coinciding with the possible states of classical *binary digits* (*bits*) [35]. Such states, or vectors, are normally represented in *Dirac notation*, also known as *bra-ket notation*, as $|0\rangle$ and $|1\rangle$ [36]. They can be represented geometrically by using the so-called *Bloch sphere* [37] [38].

Unlike traditional ones, all gates used in quantum computing must be *reversible*: this means that the original information can always be reconstructed. To clarify this concept, if we perform a subtraction, for example $9 - 4 = 5$, it is not possible to return to the operands starting from the result; a loss of information then occurs. Examples of some gates normally used in quantum computing are the following [39]:

- The *NOT* gate flips the state of a qubit from $|0\rangle$ to $|1\rangle$ and vice versa. It is graphically represented as a rounded block with a + on it or as a square block with an X on it.
- The *CNOT* gate, which is also called the *controlled-not* gate, flips the state of a *target qubit* from $|0\rangle$ to $|1\rangle$ and vice versa if the value of a second qubit, called *control qubit*, is set to $|1\rangle$. It is graphically represented as a rounded block with a + near the target qubit and a line ending in a circle to represent the conjunction with the control qubit.
- The *Toffoli* gate, which is also called the *controlled-controlled-NOT* gate, flips the state of a *target qubit* from $|0\rangle$ and $|1\rangle$ and vice versa if the value of two *control qubits* is set to $|1\rangle$. It can be used to implement NOT, AND, and XOR operations. It is graphically represented as a rounded block with a + near the target qubit and a line with two circles to represent the conjunction with the two control qubits.
- The *Hadamard* gate puts the state of a qubit in an equal superposition; this means that the state of the qubit has the same probability of being $|0\rangle$ and $|1\rangle$. It offers the simplest way to achieve true randomness as discussed in the previous section. It is graphically represented as a square block with an H on it.
- The *SWAP* gate, as the name implies, swaps the state of two qubits. It is graphically represented as a line with two X at the ends.
- The *RX* gate performs a rotation of the qubit state around the x-axis by a certain angle. It is graphically represented as a square block on which *RX* and possibly the rotation value are written.
- The *RY* gate, like the *RX* gate, performs a rotation of the qubit state around the y-axis by a certain angle. It is graphically represented as a square block on which *RY* and possibly the rotation value are written.
- The *RZ* gate, like the *RZ* and *RY* gate, performs a rotation of the qubit state around the z-axis by a certain angle. It is graphically represented as a square block on which *RZ* and possibly the rotation value are written.

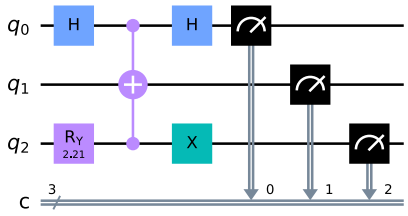


Fig. 2. One of the possible quantum circuits that can be used to generate an equiprobable number between 1 and 5.

While a classical bit value can be read at any time, the operation of reading a qubit, called *measurement*, destroys the coherence of the qubit itself and makes it collapse to a basis state. For this reason, the *measurement* operation is usually performed at the end of several classical and quantum operations on one or more qubits to extrapolate the result. All the quantum gates and operations previously described can be seen in Fig. 1.

Today's quantum processors are similar in size to their classic counterparts. To preserve their properties, they operate in an isolated environment and at very low temperatures, close to absolute zero. This temperature is theoretically the lowest that can be reached according to the law of physics [14] [41].

C. Quantum Multi-Bit Adders

The simplest way to add two bits is to use a *half-adder*. Given two inputs A and B, it returns their sum and the carryover. The problem with this adder is that it doesn't handle the previous carry, so it can't be stacked.

For this reason, for *multi-bit adders* the basic reference element is the *full-adder*, which unlike the *half-adder* also manages the previous carry. Stacking multiple *full-adders* allows for *multi-bit adders*. Depending on how the carry is handled during the bitwise addition, *multi-bit adders* can be divided into different types. It is possible to implement such adders also in quantum computing. The most commonly used types of multi-bit *quantum adders* are the following:

- The *Quantum Ripple-Carry Adder* (*QRCA*) [42] [43] [44] is the simplest of all adders. It calculates the carry of each bit addition and inputs it into the next bit addition.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Games. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TG.2023.3251663

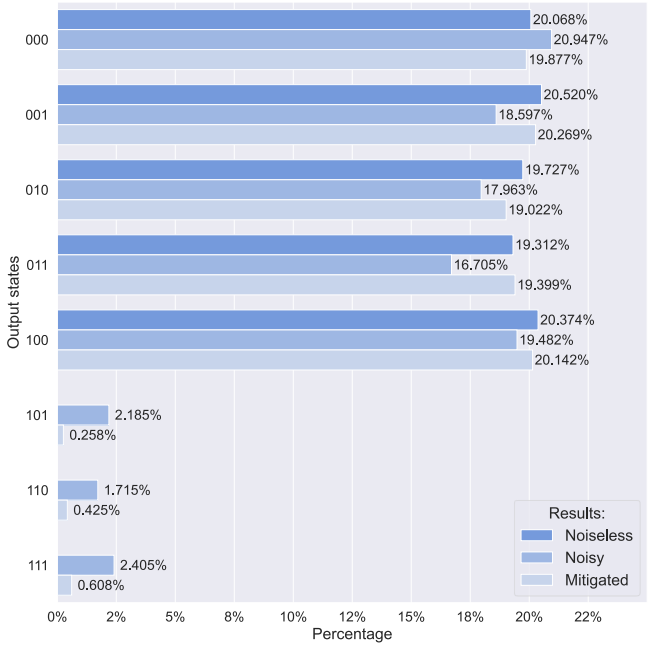


Fig. 3. Comparison of the results obtained by running *the five-values number generator circuit* for 16384 shots first on the flat IBM® QASM simulator (*Noiseless*) and then applying to it the noise model of IBM® Quito (*Noisy*), a real quantum computer. These last measurements were then further refined through calibration techniques (*Mitigated*) to remove, as far as possible, the unwanted states.

- The *Quantum Carry-Save Adder* (*QCSA*) [45] calculates subtotals and carry-over of multiple inputs. They will subsequently be used for the calculation of the final sum.
- The *Quantum Carry-Lookahead Adder* (*QCLA*) [46] [47] [48] improves *QRCA* by solving the carry propagation problem at the cost of major complexity. It is currently the fastest-known technique for performing additions, especially with many bits.

III. Quantum Implementation Of The Morra Game

According to the definition of quantum game given in [17], the implementation of the Quantum Morra game and the variants discussed in this paper has firmly recoursed to the dimension of quantum technologies for its development and actively uses it also for its functioning. This section will then provide a complete description of the making of the game.

A. Tools Used

IBM® Qiskit®, a leading open-source Software Development Kit (SDK) for the realization of quantum computing solutions, was used for the study and the realization of the circuits [49].

In Qiskit®, every single run of a circuit is called a shot. This terminology, now in common use in quantum computing, will also be used in this paper.

B. Generation of Equiprobable Quantum Numbers with Five Values

The first step to creating a quantum random player for the

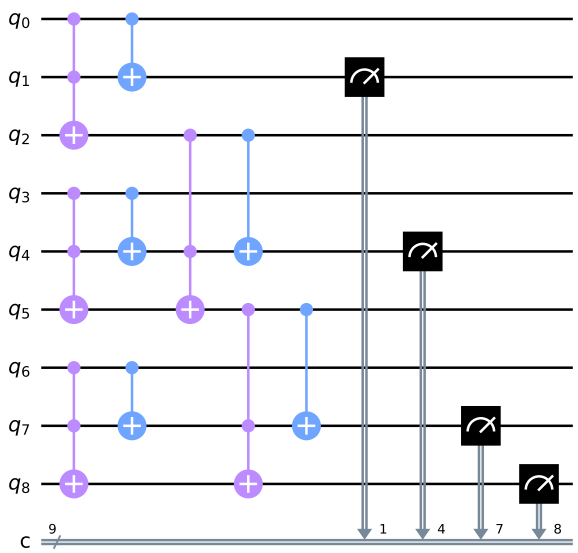


Fig. 4. The adder circuit implemented for the Morra game, in where is necessary to calculate the sum of two numbers representable with three qubits.

Morra game consists in realizing a quantum circuit to extract the first number, which represents the number mimicked by the human hand, and a second number necessary to calculate the supposed sum.

Previously it was said that, just like the bits of classical computers, once a qubit is measured it can have two possible states: 0 or 1. For this reason, also for quantum computers base 2 is the reference one. This means that, given n qubits, it is possible to represent up to 2^n different values (states). For this reason, implementing a true randomizer circuit for powers of two numbers is very simple, since for each bit a *Hadamard* gate and a *measurement* operation are sufficient. However, things got a little complicated when it is needed to generate a number in a different interval. In these cases, it is necessary to reduce the probability that certain states can be reached to zero. This is generally achieved with rotations, which can induce phase changes.

In the case of the Morra game, a number between 1 and 5 must be extracted, desirably with the same probability. This means that it is possible to extract a total of 5 different numbers. Since 5 is not a power of two, to obtain such a number is necessary to take the power of two immediately higher, which is 8, and let the probability of reaching $8 - 5 = 3$ possible states vanish. In this way, only numbers between 1 and 5 can be extracted. One of the possible circuits capable of obtaining this result is depicted in Fig. 2. It is composed of:

- An *RY* gate, which rotates the qubit q_2 by $\pi/1.4187762$ radians.
- A *Toffoli* gate, which sets q_1 to 1 only if q_0 and q_2 are equal to 1.
- Two *Hadamard* gates, of which the first put q_0 in equal superposition while the second resets its value to 0 with a probability of 100%. The superposition of q_0

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Games. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TG.2023.3251663

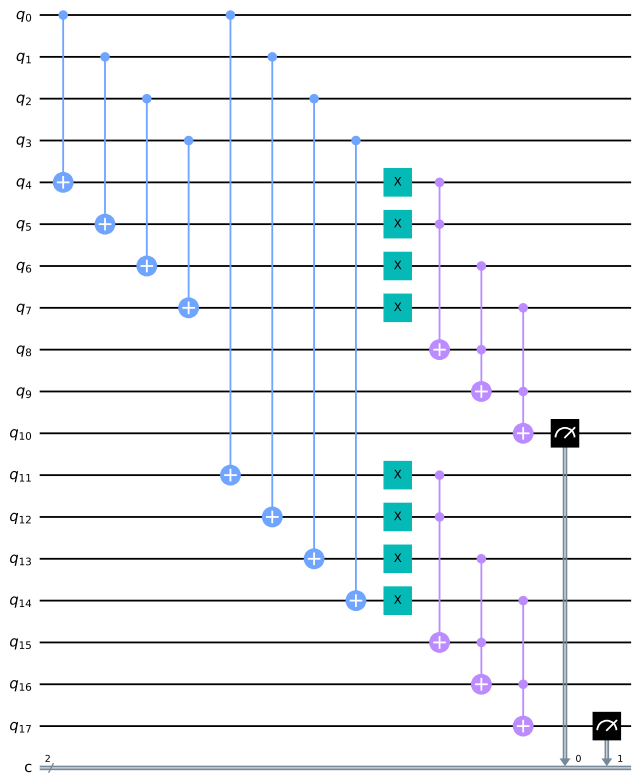


Fig. 5. A screenshot of the comparator circuit, used to compare the supposed sum of the players with the correct one.

allows triggering the *Toffoli* gate.

- A *NOT* gate, which flips the value of q_2.
- Three *measurement* operations, which are necessary to obtain the final value of each qubit.

This circuit was obtained by using a Monte Carlo technique. Several tests were carried out to find a circuit capable of canceling 3 of the 8 output states. Once found, it has been optimized to reduce the number of gates and to have near output states. Subsequently, a brute force approach was applied to find the exact *RY* rotation that could lead to the same probability (20%) for each state. The final circuit can generate numbers between 0 and 4 and will be referred to below as the *five-values number generator circuit*.

C. Analysis of the Results

The *five-values number generator circuit* was run for 16384 shots two times on the IBM® QASM simulator. In the first run there was no noise (*Noiseless)*, while in the second was added the noise model of IBM® Quito, a real 5-qubit quantum computer (*Noisy*). The outcome of the execution can be seen in Fig. 4. It is immediately visible that, unlike the optimal results obtained by the IBM® QASM simulator in the absence of noise, 8 values are returned instead of the expected 5. This is due to the fact that quantum computers are not perfect machines. To function optimally, they should be completely isolated from the surrounding environment. On classical computers, supplying one or more inputs to a program result in correspondingly one or more outputs. The problem with quantum computers is that

```
QUANTUM MORRA
Welcome to the Quantum Morra game!
The game will end once 16 points have been reached

Please enter 5 if you want to play with 5 numbers (vanilla version)
or 6 if you to play with 6 numbers (allowing zero variant): 5

ROUND 1
Digit the hand mimicked number: 5
Digit the supposed sum: 7
Calculation of the Quantum Player number...
Job Status: job has successfully run
QUANTUM PLAYER NUMBER: 2
Calculation of the Quantum Player supposed sum...
Job Status: job has successfully run
QUANTUM PLAYER SUPPOSED SUM: 5
Addition and comparison of the supposed sums...
Job Status: job has successfully run
COMPARISONS: 01

Player supposed the correct sum!
Points - Player: 1 , Quantum Player: 0


ROUND 2
Digit the hand mimicked number:
```

Fig. 6. A screenshot of the Quantum Morra game implementation. At the start of the game, it allows the user to choose whether to allow zero or not.

their isolation property will be implicitly lost if it is wanted that their behavior varies based on the inputs provided. Due to this reason, despite continued efforts to improve this property, they will continue to suffer from noise and decoherence problems. Then, quantum error correction and mitigation algorithms are critical for improving the result of the calculations [50] [51].
One of the possible ways to improve this result is to perform a measurement calibration. This is possible by preparing the 2^n basis input states and evaluating the probability of counting obtainable by measuring in another basis state. Since the *number generator circuits* analyzed in this paper uses three qubits, there will be $2^3 = 8$ total possible quantum states. Using the IBM® QASM simulator, it is possible to calculate the calibration matrix in the absence of noise: in this particular case, it will be a simple identity matrix. By applying the noise model of a real quantum computer on the IBM® QASM simulator, such as that of IBM® Quito, it is possible to calculate the calibration matrix for this quantum computer to mitigate its measurement error. The *Mitigated* results obtained by applying this technique are always shown in Fig. 3. The improvement in the results is clearly visible.

B. Adder for the Morra Game

Now that the circuit for generating equiprobable random numbers in the desired range was found, it is necessary to build an adder for:

- Generating the supposed sum of the quantum random player.
- Calculating the effective sum for comparing it with the sums supposed by the players.

The numbers to be added in the Morra game can be contained in three bits. For this reason will be used a simple *multi-bit adder*, depicted in Fig. 4. Since current quantum computers

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Games. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TG.2023.3251663

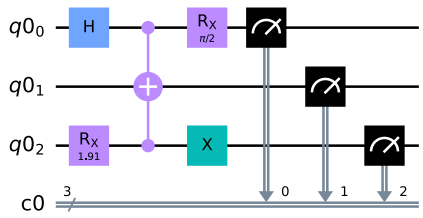


Fig. 7. One of the possible quantum circuits that can be used to generate an equiprobable number between 0 and 5.

have a limited number of qubits available, a quantum circuit that mixes a *half-adder* for the first qubit and two *full-adders* for the remaining qubits has been used. In this way, to add two 3-bit numbers only 9 qubits are needed. Given two 3-bit numbers A and B, starting from the Most Significant Bit (MSB) the bits should be input into the circuit as follows:

- $A[2] \rightarrow q_6,\ A[1] \rightarrow q_3,\ A[0] \rightarrow q_0.$
- $B[2] \rightarrow q_7,\ B[1] \rightarrow q_4,\ B[0] \rightarrow q_1.$

Following the calculation, the output bits with a higher index will contain the MSBs of the resulting sum. So it will read as follows: $O[3] \rightarrow q_8,\ O[2] \rightarrow q_7,\ O[1] \rightarrow q_4,\ O[0] \rightarrow q_1.$

E. Comparison of the supposed sums with the correct one

To obtain the complete quantum version of the Morra game, a final circuit is needed to compare the two supposed sums by the players with the effective sum. Depending on their value:

- They can both be different from the effective one.
- One of them can be equal to the effective one.
- Both can be equal to the effective one.

The simplest circuit for comparing two qubits can be obtained simply using a *CNOT* gate and measuring the *target qubit*: if the qubits have the same value the output is 0, vice versa the output will be 1. By extending the same principle, it is therefore possible to use 12 qubits to compare the 4 *control* qubits of the effective sum (C) with the respective 8 *target* qubits of the supposed sums (S_1, S_2). Starting from their MSBs, the sums are input into the circuit in this way:

- $C[3] \rightarrow q_0,\ C[2] \rightarrow q_1,\ C[1] \rightarrow q_2,\ C[0] \rightarrow q_3$
- $S_1[3] \rightarrow q_4,\ S_1[2] \rightarrow q_5,\ S_1[1] \rightarrow q_6,\ S_1[0] \rightarrow q_7$
- $S_2[3] \rightarrow q_{14},\ S_2[2] \rightarrow q_{13},\ S_2[1] \rightarrow q_{12},\ S_2[0] \rightarrow q_{11}$

After this step, a downsampling mechanism can be applied to measure the output of two single qubits, which are sufficient to represent the previously mentioned states. First, 8 *NOT* gates, 4 for every single comparison, are used to flip the values of the 8 *target qubits*. This operation is useful for restoring the $0s$ to $1s$ after the previous operation. Second, 6 *Toffoli* gates, 3 for every single comparison, are used to downsample the 8 qubits to 2 qubits, 1 for every single comparison. The output of this downsampling operation will be 1 if all previous qubits were equal to 1, i.e. the two numbers being compared are equal, otherwise it will be 0.

The final comparator circuit capable of carrying out the above operation is shown in Fig. 5.

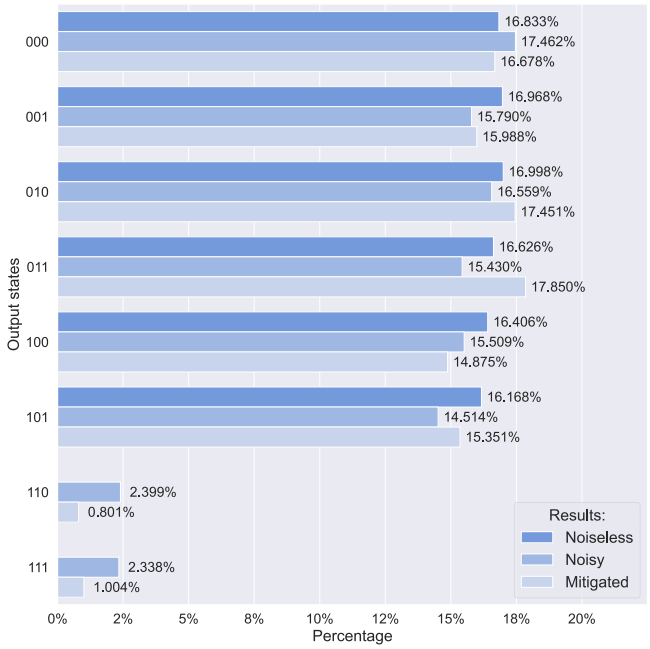


Fig. 8. Comparison of the results obtained by running the *six-values number generator circuit* for 16384 shots first on the flat IBM® QASM simulator (*Noiseless*) and then applying to it the noise model of IBM® Quito (*Noisy*), a real quantum computer. These last measurements were then further refined through calibration techniques (*Mitigated*) to remove, as far as possible, the unwanted states.

F. Final Circuit

The final circuit was obtained by stacking two *number generator circuits* and the *adder* discussed above. A text-based interface has also been added to allow the user to play against the aforementioned quantum random player until one of them wins. An image of the final game is shown in Fig. 6. In this version, during each round the user must type in two numbers: the value mimicked with the hand and the supposed sum he will reach with the number thrown by the quantum random player. At this point, the quantum random player will do the same and, after making its choices, the final calculation will be made to verify the winner. The number of points needed to win is by default set at 16 as in the Sardinian Morra rules, but this value can be changed freely.

Along with this document, the complete Python code for the Quantum Morra is provided. An *IBMid* is required to obtain a new *API token* as it uses the *IBM® Quantum services* [52], which are necessary to play the game.

IV. Variants of the Morra Game with Zero Allowed

Some variants of the Morra game allow players to use the number zero, which can be mimicked by simply leaving the hand closed. These variants change both the total number of extractable numbers, which goes from 5 to 6, and the sum that players can suppose, which goes from a range between 2 and 10 to one between 0 and 10. While the second point does not affect the adder circuit, the first requires creating a new *number generator circuit*, which will be discussed next.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Games. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TG.2023.3251663


```
QUANTUM ODDS AND EVENS
Welcome to the Quantum Odds And Evens game!
The game will end once 3 rounds have been reached

Please enter 5 if you want to play with 5 numbers (vanilla version)
or 6 if you to play with 6 numbers (allowing zero variant): 5
Please enter 0 to assign you the even numbers or 1 to assign you the odd numbers: 1

ROUND 1
Digit the hand mimicked number: 3
Calculation of the Quantum Player number...
Job Status: job has successfully run
QUANTUM PLAYER NUMBER: 5
Calculation of the sum of the previously drawn numbers...
Job Status: job has successfully run
ADDITION: 8

The number is EVEN!
Quantum Player scores one point!


ROUND 2
Digit the hand mimicked number:
```

Fig. 9. A screenshot of the Quantum Odds and Evens game implementation. At the start of the game, it allows the user to choose whether to allow zero or not.

A. Six-Values Equiprobable Number Generator Circuit

There are different ways to create a circuit capable of extracting a number between 0 and 5 with the same probability. Fig. 7 shows one that bears a strong resemblance to the one shown in Fig. 3. In fact, it was obtained starting from it and making the following substitutions:

- One of the *Hadamard* gates has been replaced with an *RX* gate, which rotates the qubit q_0 by $\pi/2$.
- The *RY* gate has been replaced with an *RX* gate, which rotates the qubit q_2 by $\pi/1.4187762$ radians.

In the previous *number generator circuit*, the first *Hadamard* gate acts on the qubit q_0 to put it in equal superposition and activate the *Toffoli* gate if the qubit q_2 is also equal to 1. The second *Hadamard* gate then returns the value of the qubit q_0 to 0 with a probability of 100%. In the novel circuit, however, the use of a single *Hadamard* gate followed by an *RX* gate does not remove the superposition state of the qubit q_0 and this creates an additional state, which is exactly what was wanted to achieve. As with the previous circuit, the exact *RX* gate rotation was found with a brute force approach to obtain a probability of 16.66667% for each output state.

B. Results

Like the previous *number generator circuit*, also this one was run for 16384 shots on the IBM® QASM simulator with and without the noise model of IBM® Quito. The calibration matrix was calculated to mitigate the unwanted effects due to noise and improve the results. They can be seen and compared in Fig. 8.

C. Final Circuit

The final circuit was obtained similarly as previously discussed for the Morra game. Also for this variant, the full Python code is provided alongside this paper.

V. Odds and Evens

Odds and Evens is a known variant of the Morra game. It was known both by the Greeks with the term “artiazein” and by the Romans with the term “ludere par impar” [53] [54]. A late medieval reference, dated around 1300 AD, has been found in Hugh von Trimberg's “Der Renner” [55]. Nowadays, it continues to be played mostly by children in different parts of the world. It is known by several names, including “bucking up”, “choosies”, and “pick”.

A. Rules

Odds and Evens rules are simpler than those of the Morra game. At the beginning of the game, the two players decide by mutual agreement who will be awarded points for the odd numbers and, consequently, for the even ones. Using a synchronization mechanism based on shouted words such as "One, two, three, shoot!" or similar, both players quickly and simultaneously mimic two numbers between 0 and 5 or 1 and 5 depending on whether zero is allowed or not. If the sum of the two mimicked numbers is odd, one point will be awarded to the odd player and vice versa. Players can choose to end the game after only one round, the best of three rounds, or after several rounds.

B. Implementation

The implementation of this variant of the Morra game does not require any additional circuits compared to those seen previously. The quantum random player can be implemented using either the *five-values* or the *six-value number generator circuits.* Then, the values chosen by the actual player and the quantum random player can be added using the *quantum adder* described above. Finally, parity can be verified by simply checking the Least Significant Bit (LSB) of the sum: if it is equal to 0 the number is even, while if it is equal to 1 the number is odd. An image of the final game, which performs these exact steps, is shown in Fig. 9. Also for Odds and Evens, the full Python code is provided alongside this paper.

VI. Conclusions

This paper discussed the implementation of a quantum random player to solve the Morra game and some of its variants in quantum computing. For the implementation of the classic Morra game and the variant with the zero allowed, such a player relies respectively on the use of a *five-values* or a *six-values number generator circuit*, needed to generate the base number and an additional number used for the calculation of the presumed sum, as well as an *adder* that performs this operation. On the other hand, for the odds and evens game, a single *five-values* or *six-values number generator* circuit and a *quantum adder* are sufficient. The code of the classic version of the Morra and the variants presented therein is provided alongside this paper. In this way, everyone can try the game with a minimum setting of the Python environment. In the future, the authors want to further their research into the possibility of adapting other games into the world of quantum computing.

References

[1] G. Ebers, “Egypt: Descriptive, historical, and picturesque”, London, England: Cassell & Co., 1878, p.166.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Games. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TG.2023.3251663

[2] E. Falkener, "Games Ancient and Oriental, and How to Play Them: Being the Games of the Greek, the Ludus Latrunculorum of the Romans and the Oriental Games of Chess, Draughts, Backgammon and Magic Squares", London, England: Longmans, Green & Co., 1892.
[3] P. F. Perdrizet, The game of Morra, J. Hellen. Stud., vol. 18, pp. 129–132, 1898, doi: 10.2307/623718.
[4] C. T. Lewis and C. Short, "mico" definition in "A Latin Dictionary". Oxford, England: Clarendon Press, 1879.
[5] M. T. Cicero, "De Officiis. With An English Translation by Walter Miller", London, England: W. Heinemann, 1913.
[6] H. U. Vogel and G. N. Dux, "Concepts of Nature: A Chinese-European Cross-Cultural Perspective", Leiden, The Netherlands: Brill, 2010, doi:10.1163/ej.9789004185265.i-566.138.
[7] "huá quán" English translation, Yabla, Chinese English Pinyin Dictionary. [Online]. Available: https://chinese.yabla.com/chinese-english-pinyin-dictionary.php?define=+%E5%88%92%E6%8B%B3+.
[8] "cāi quán" English translation, Yabla, Chinese English Pinyin Dictionary. [Online]. Available: https://chinese.yabla.com/chinese-english-pinyin-dictionary.php?define=+%E7%8C%9C%E6%8B%B3+.
[9] "mǔ zhàn" English translation, Yabla, Chinese English Pinyin Dictionary. [Online]. Available: https://chinese.yabla.com/chinese-english-pinyin-dictionary.php?define=+%E6%8B%87%E6%88%98+.
[10] Xie Zhaozhe, "Fivefold Miscellany (Wuzazu)", pp. 500-501, 1608.
[11] F. Delogu, M. Barnewold, C. Meloni, E. Toffalini, A. Zizi, and R. Fanari, "The Morra Game as a Naturalistic Test Bed for Investigating Automatic and Voluntary Processes in Random Sequence Generation", Frontiers in Psychology, vol. 11, 2020, doi: 10.3389/fpsyg.2020.551126.
[12] Sa Murra game, Davide Onida. [Online]. Available: http://samurra.it/.
[13] AGI editorial staff, "Nessuno batte i sardi a morra, nemmeno un robot", AGI - Agenzia Giornalistica Italia, July 2021. [Online]. Available: https://www.agi.it/cronaca/news/2021-07-25/sardegna-robot-perde-contro-umani-gioco-morra-13377028/.
[14] IBM, "What is quantum computing?," IBM. [Online]. Available: https://www.ibm.com/quantum-computing/what-is-quantum-computing/.
[15] D. Frauchiger, R. Renner and M. Troyer, "True randomness from realistic quantum devices," ArXiv, 2013, arXiv:1311.4547v1.
[16] M. Gordon and G. Gordon, "Quantum computer games: Quantum minesweeper", Physics Education, vol. 45, no. 4, 2010, doi: 10.1088/0031-9120/45/4/008.
[17] L. Piispanen, M. Pfaffhauser, A. Kultima, J. R. Wootton, "Defining Quantum Games", arXiv: Quantum Physics, 2019, arXiv: 2206.00089.
[18] J. Wootton, "Introducing the world's first game for a quantum computer", Medium, 2017. [Online]. Available: https://decodoku.medium.com/introducing-the-worlds-first-game-for-a-quantum-computer-50640e3c22e4.
[19] J. Wootton, "Quantum Battleships: The first game for a quantum computer", Medium, 2017. [Online]. Available: https://decodoku.medium.com/quantum-battleships-the-first-multiplayer-game-for-a-quantum-computer-e4d600ccb3f3.
[20] S. G. Akl, "On the importance of being quantum," Parallel Processing Letters, vol. 20, no.3, 2010, pp. 275-286, doi: 10.1142/S0129626410000223.
[21] C. Cantwell, "Quantum Chess: Developing a Mathematical Framework and Design Methodology for Creating Quantum Games", arXiv: Quantum Physics, 2019, arXiv: 1906.05836.
[22] A. Anand, B. K. Behera, and P. K. Panigrahi, "Solving diner's dilemma game, circuit implementation and verification on the IBM quantum simulator", Quantum Information Processing, vol. 19, no. 186, 2020, doi: 10.1007/s11128-020-02687-5.
[23] K. Becker, "Flying Unicorn: Developing a Game for a Quantum Computer", arXiv: Quantum Physics, 2019, arXiv:1910.08238
[24] A. Ranchin, "Quantum Go", arXiv: Quantum Physics, 2016, arXiv: 1603.04751
[25] L. Qiao et al., "Quantum Go Machine", arXiv: Quantum Physics, 2020, arXiv:2007.12186
[26] C.-F. Li, Y.-S. Zhang, Y.-F. Huang, G.-C. Guo, "Quantum strategies of quantum measurements," Physics Letters A, vol. 280, no. 5–6, 2001, pp. 257-260, doi: 10.1016/S0375-9601(01)00072-X.
[27] A. P. Flitney and D. Abbott, "Quantum version of the Monty Hall problem", Physical Review A, vol. 65, no. 6, 2002, doi: 10.1103/PhysRevA.65.062318
[28] L. F. Quezada and S.-H. Dong, "Quantum Version of a Generalized Monty Hall Game and Its Possible Applications to Quantum Secure Communications", Annalen der Physik 2020, vol. 533, no. 1, 2020, doi: 10.1002/andp.202000427.
[29] F. G. Fuchs, V. Falch and C. Johnsen, "Quantum Poker—a game for quantum computers suitable for benchmarking error mitigation techniques on NISQ devices,". The European Physical Journal Plus, vol. 135, no. 353, 2020, doi: https://doi.org/10.1140/epjp/s13360-020-00360-5
[30] A. Pal, S. Chandra, V. Mongia, B. K. Behera, and P. K. Panigrahi, "Solving Sudoku game using a hybrid classical-quantum algorithm", EPL, vol. 128, no. 4, 2019, doi: 10.1209/0295-5075/128/40007.
[31] T. Glasgow, H. Hilton, R. Brantley, and O. Levy, "Quantum Tetris", 2020. [Online]. Available: https://github.com/dartmouth-cs98/Quantum-Tetris.
[32] W. Xiao, E. Lari, B. Ho, S. Parekh, and O. Brückner, "Quantum Tetris", 2021. [Online]. Available: https://olivierbrcknr.github.io/quantum-tetris/.
[33] A. Goff, D. Lehmann and J. Siegel. "Quantum Tic-Tac-Toe, Spooky-Coins & Magic-Envelopes, as Metaphors for Relativistic Quantum Physics," AIAA 2002-3763. 38th AIAA/ASME/SAE/ASEE Joint Propulsion Conference & Exhibit. July 2002.
[34] J. N. Leaw and S. A. Cheong, "Strategic Insights From Playing the Quantum Tic-Tac-Toe", Journal of Physics A: Mathematical and Theoretical, vol. 43, no. 45, 2010, doi: 10.1088/1751-8113/43/45/455304.
[35] Azure Quantum Documentation, "The qubit in quantum computing," Microsoft. [Online]. Available: https://docs.microsoft.com/en-us/azure/quantum/concepts-the-qubit.
[36] Azure Quantum Documentation, "Dirac notation," Microsoft. [Online]. Available: https://docs.microsoft.com/en-us/azure/quantum/concepts-dirac-notation.
[37] F. Bloch, "Nuclear induction," Phys. Rev., vol. 70, no. 7-8, pp. 460–474, Oct. 1946, doi: 10.1103/physrev.70.460.
[38] F. T. Arecchi, E. Courtens, R. Gilmore and H. Thomas, "Atomic Coherent States in Quantum Optics," Phys. Rev. A, vol. 6, no. 6, pp. 2211-2237, Dec. 1972, doi: 10.1103/PhysRevA.6.2211.
[39] IBM, "Operations glossary," IBM. [Online]. Available: https://quantum-computing.ibm.com/composer/docs/iqx/operations_glossary.
[40] IBM, "IBM Quantum Composer," IBM. [Online]. Available: https://quantum-computing.ibm.com/composer/.
[41] D-Wave, "Welcome to D-Wave: D-Wave's Quantum Computer Systems," D-Wave. [Online]. Available: https://docs.dwavesys.com/docs/latest/c_gs_1.html#d-wave-s-quantum-computer-systems.
[42] V. Vedral, A. Barenco, and A. Ekert, "Quantum networks for elementary arithmetic operations," Physical Review A, vol.54, no.1, July 1996, doi: 10.1103/PhysRevA.54.147.
[43] S. A. Cuccaro, T. G. Draper, S. A. Kutin and D. P. Moulton, "A new quantum ripple-carry addition circuit," arXiv: Quantum Physics, 2004, arXiv:quant-ph/0410184
[44] F. Wang, M. Luo, H. Li, Z. Qu and X.Wang, "Improved quantum ripple-carry addition circuit," Science China Information Sciences, vol. 59, no. 042406, 2016, doi:10.1007/s11432-015-5411-x
[45] P. Gossett, "Quantum carry-save arithmetic," arXiv: Quantum Physics, 1998, arXiv:quant-ph/9808061.
[46] T. G. Draper, S. A. Kutin, E. M. Rains and K. M. Svore, "A logarithmic-depth quantum carry-lookahead adder", Quantum Information & Computation, vol. 6, no. 4, pp. 351-369, 2006.
[47] A. Trisetyarso and R. Van Meter, "Circuit Design for A Measurement-Based Quantum Carry-Lookahead Adder," International Journal of Quantum Information, vol. 8, no. 5, pp. 843-867, 2010, doi: 10.1142/S0219749910006496.
[48] H. Thapliyal, E. Muñoz-Coreas, and V. Khalus, "Quantum circuit designs of carry lookahead adder optimized for T-count T-depth and qubits," Sustainable Computing: Informatics and Systems, vol. 29 part B, no. 100457, 2021, doi: 10.1016/j.suscom.2020.100457.
[49] M. S. Anis et al., "Qiskit: An Open-source Framework for Quantum Computing,", IBM, 2021. [Online]. Available: https://qiskit.org/.
[50] P.W. Shor, "Quantum computing," Documenta Mathematica Extra Volume ICM, vol.1, 1998, pp. 467-486.
[51] M. Chiani and L. Valentini, "Short Codes for Quantum Channels With One Prevalent Pauli Error Type," IEEE Journal on Selected Areas in Information Theory, vol. 1, no. 2, pp. 480-486, Aug. 2020, doi: 10.1109/JSAIT.2020.3012827.
[52] IBM, "IBM Quantum," IBM. [Online]. Available: https://quantum-computing.ibm.com/.
[53] Lorenzo Lippi, "Malmantile racquistato", Florence, Italy: S.A.S alla Condotta, 1688.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Games. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TG.2023.3251663

[54] W. Smith, W. Wayte and G. E. Marindin. "A Dictionary of Greek and Roman Antiquities", London, England: John Murray, 1890.
[55] H. Von Trimberg, "Der Renner", G. Ehrismann, Ed., Stuttgart, Germany: Litterarischer Verein, 1908.



Antonio Costantino Marceddu received the B.Sc. and the M.Sc. in computer engineering from Politecnico di Torino, respectively in 2015 and 2019.

From 2020, after a short time as a Research Assistant, he is a Ph.D. student of the Politecnico di Torino under the supervision of Professor Bartolomeo Montrucchio. His research interests include machine learning, computer vision, computer graphics, and quantum computing.



Bartolomeo Montrucchio received the M.Sc. degree in electronic engineering and the Ph.D. degree in computer engineering from Politecnico di Torino, Turin, Italy, in 1998, and 2002, respectively.

He is currently a Full Professor of Computer Engineering with the Dipartimento di Automatica e Informatica, Politecnico di Torino. His current research interests include image analysis and synthesis techniques, scientific visualization, sensor networks, RFIDs, and quantum computing.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/