# Simulating Quantum Turing Machine in Augmented Reality

Wanwan Li
Email:wanwan@usf.edu
University of South Florida
Tampa, Florida, USA

## ABSTRACT

As quantum computing theory is attracting attention from researchers nowadays, visualizing the quantum computing process is necessary for fundamental quantum computing education and research. Especially, connecting traditional computational theory with advanced quantum computing concepts is an extremely important step in learning and understanding quantum computing. In this paper, we propose a practical interactive interface for simulating Quantum Turing Machine (QTM) in Augmented Reality (AR) that combines the traditional Turing machine computational model with the quantum computing simulation. Through such an interface, users can use a C-like script to represent a QTM and simulate such QTM in an immersive augmented reality platform through the Vuforia AR engine. After validating our proposed QTM AR simulator through a series of experiments, we show the great potential to apply our QTM AR simulator to quantum computing education through a realistic and interactive visualization interface in augmented reality.

## CCS CONCEPTS

**• Computing methodologies → Graphics systems and interfaces**; **Quantum Computing**; **• Human-centered computing → Systems and tools for interaction design**.

## KEYWORDS

Computational Theory, Augmented Reality, Quantum Turing Machine (QTM)

**ACM Reference Format:**
Wanwan Li. 2023. Simulating Quantum Turing Machine in Augmented Reality. In *Proceedings of 8th International Conference on Multimedia and Image Processing (ICMIP 2023).* ACM, New York, NY, USA, 6 pages. https://doi.org/XXXXXXX.XXXXXXX

## 1 INTRODUCTION

As a fundamental computational model for modern digital computers that are based on Von Neumann architecture (with main components including CPU, Memory, and IO system, etc.) [9], Alan Turing first propose a computable automaton named after himself, i.e., Turing machine [11]. Turing machine is a mathematical model of computation describing an abstract machine that manipulates symbols on a strip of tape according to a table of rules which is capable of implementing any computer algorithm. On the other hand, with the development of modern physics which is built upon the theory of relativity and the theory of quantum mechanics [4], researchers realize the promising application of quantum entanglement for the next generation of supercomputers, i.e., quantum computers [12]. Therefore, given these observations, we proposed the first practical interactive interface for simulating Quantum Turing Machine (QTM) in Augmented Reality (AR) that combines the traditional Turing machine computational model with the quantum computing simulation. More specifically, we propose a C-like script that helps users to encode a QTM which can be interpreted by our AR simulator so as to simulate the computational process of a QTM in an immersive augmented reality platform through the Vuforia AR engine. Figure 1 shows an example of our QTM AR simulator where a user is interacting with a QTM in an AR environment through a mobile device which can be applied in quantum computing education.

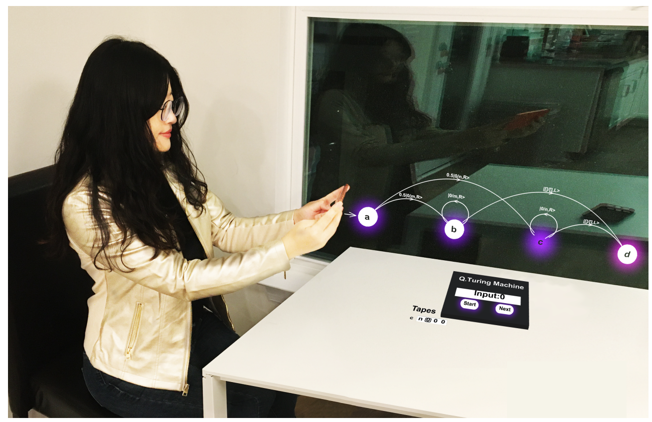



**Figure 1: AR Quantum Turing Machine Simulator.**

**Quantum Computing.** In 1980, when Richard Feynman was trying to design a computer program to simulate quantum events, due to the limitation of the computation power of the existing computers and the fact that computation time complexity increases exponentially as the number of subatomic particles is increased, Richard Feynman realized that traditional computational algorithms are not efficient enough to handle the growing complexity of quantum calculations. Therefore, Richard Feynman suggests that this challenging simulation problem can be solved by employing the quantum elements themselves to accomplish the computation tasks, i.e., by building a bridge between quantum physics and computer science. This is the original high-level idea of quantum computing. As the basis for quantum computing, it is an essential step to understanding what is a quantum bit, for short, a qubit, which is usually taken as a counterpart of a classical bit. In quantum mechanics, before a measurement was taken, subatomic particles exist in a state called superposition. This phenomenon also applies to

Place the footnote text for the author (if applicable) here.
*ICMIP 2023, April 21–23, 2023, Tianjin, China*

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/XXXXXXX.XXXXXXX

a qubit. Different from a classical bit of a binary value that can only represent a value of 0 or 1, a qubit that is made of subatomic particles can represent a value of 0, 1, or a random linear combination between 0 and 1. Before a measurement was taken, there is a probability distribution of a cubit to be 0 or 1 given the amplitudes. Therefore, before the measurement, a qubit can represent both 0 and 1 at the same time. This means, a qubit can store much more information than a classical bit. After the measure, the superposition state of a qubit collapse to a classical and determined state of either 0 or 1 to get an output. Figure 2 shows the relation between classical bit and qubit through an example that illustrates how a qubit is measured. When the qubit $\psi$ is unmeasured, it is a superposition of $|\psi\rangle = 0.3|1\rangle + 0.7|0\rangle$ with an expectation values of $\langle\psi\rangle = 0.3 \times 1 + 0.7 \times 0 = 0.3$ . After the measurement, qubit has a 30% chance to collapse into a 1 and a 70% chance to become a 0.

**Turing Machine Simulator.** Turing machine is the mathematical model in automaton theory that contains a central processing unit (CPU) controlling all data manipulation done by a computer, with the canonical device using sequential memory to store data. More specifically, it is an automaton capable of enumerating some arbitrary subset of valid strings that are part of a recursively enumerable set. Turing machine has a tape of infinite length on which it can perform read and write operations. Turing machine simulator is a program that simulates the computation process on a Turing machine that either accepts or rejects a given tape with input and output on it. For education purposes, a good design of the Turing machine simulator is essential for understanding the behavior of a Turing machine so as to help students design Turing machines to solve practical problems. Especially, for understanding the quantum computing theory through the Turing machine simulator is a novel idea and has never been explored by existing research works. More specifically, the Turing machine simulator that presents a quantum computing process on automata is called Quantum Turing Machine (QTM) simulator. However, most of the existing works on simulating automatons and Turing Machines, such as the multitape Turing machines simulator [5], probabilistic Random-Access Machine (RAM) simulator [10], left-linear rewrite rule-based Turing machine simulator [2], uEAC-Computable Functions-based Turing machine simulator [15], islands model metaphorical representation-based automaton simulator [3], Hololens-based AR Turing machine simulator [6] etc., are not considering QTM.

**QTM Simulator on AR.** A research work closely related to ours can be found in a theoretical paper written by Carpentieri et.al. [1] which theoretically validates the possibility of simulating QTA through quantum mechanics. However, there is no visualization of such a simulation process provided by that paper. As inspired by the work presented by Zable et al. [14] that is developing an effective visualization tool for quantum computing education using Virtual Reality (VR), in this paper, we propose an immersive 3D interactive interface for quantum Turing machine design and simulation in Augmented Reality (AR). Contributions of our work include:

- We propose a novel C-like programming script to describe the quantum Turing machine in plain text and implement an interpreter for this C-like script to construct an interactive quantum Turing machine that simulates arbitrary input.

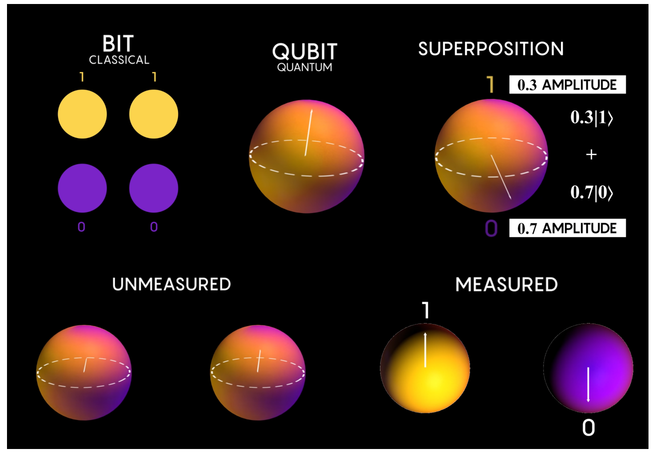



**Figure 2: Introduction to Qubit (Quantum Bit)[8].**

- We develop an AR-enabled quantum Turing machine simulator through which users can interact with the simulator through buttons on a mobile device or webcam that is supporting Vuforia AR configurations so as to help user design and simulate QTMs in AR.
- We validate our proposed AR simulator through a series of numerical experiments on testing different types of quantum automata, such as Quantum Finite Automaton (QFA), Quantum Pushdown Automaton (QPDA), and Quantum Turing Machine (QTM). Experiment results can be found through this link: https://youtu.be/hjnySWhslF0

## 2 QUANTUM AUTOMATON THEORY

Quantum automaton theory is a variation of the classic automaton theory by extending the existing automaton's states into quantum states. There are three basic types of quantum automaton including: Quantum Finite Automaton (QFA), Quantum Pushdown Automaton (QPDA), and Quantum Turing Machine (QTM). Quantum Finite Automaton (QFA) is an quantum automaton with a finite number of states that are following a list of transition rules called transition function $\delta : Q \times \Sigma \rightarrow P|Q\rangle$. Given current measured active state $q_i \in Q$ and the input alphabet $a \in \Sigma$, $q_i$ will transits to an arbitrary quantum state following this transition function:

$$\delta(q_i, a) = \sum_{j=1}^{|Q|} p_{i,j}|q_j\rangle, \tag{1}$$

where the transition probability $p_{i,j} \neq 0 \in P$. Similar to QFA, Quantum Pushdown Automaton (QPDA) is a type of quantum automaton by extending QFA with a stack so as to accept context-free grammar. For QPDA, given current measured active state $q_i \in Q$ and the input alphabet $a \in \Sigma$, it will replace stack top $b$ into new stack top $c$ where stack top $b, c \in \Gamma$ are from the stack alphabet, and $q_i$ transits to a quantum state following this transition function:

$$\delta(q_i, a) = \sum_{j=1}^{|Q|} p_{i,j}|q_j/b \sim c\rangle, \tag{2}$$

where the transition probability $p_{i,j} \neq 0 \in P$. As another type of quantum automaton, Quantum Turing Machine (QTM) consists of

| | p | q | r | f |
|---|---|---|---|---|
| p | 0.5\|a/X,R> | 0.5\|a/X,R> | | |
| q | | 0.5\|b/Y,R> | 0.5\|b/Y,R> | |
| r | | | | \|[]/[],L> |

**Figure 3: The transition table of a quantum Turing machine.**

an infinite-length tape divided into cells where there is the given input. After reading an input alphabet $a \in \Sigma$, that input will be replaced to another tape alphabet $b \in \Gamma$, its header moves from one cell to another by moving left (L) or right (R). At the same time, current measured active state $q_i \in Q$ will transit to an arbitrary quantum state following this transition function:

$$\delta(q_i, a) = \sum_{j=1}^{|Q|} p_{i,j} |q_j/b, X \in \{L, R\}\rangle, \tag{3}$$

where the transition probability $p_{i,j} \neq 0 \in P$.

## 3 IMPLEMENTATIONS

**Interpreter.** In our proposed simulator, both Quantum Finite Automaton (QFA), Quantum Pushdown Automaton (QPDA), and Quantum Turing Machine (QTM) can be represented with C language-like grammar. As shown in Figure 4, in this example, the scripts describe a QTM that accepts $a^m b^n, m, n = 1, 2, 3, \ldots$ and replace $a \rightarrow X$ and $b \rightarrow Y$. We implement an interpreter for this script that can construct the data structure of such QTM for interactive simulations. We have mainly implemented two types of operations when interpreting the input scripts. For example, when the interpreter is extracting the data structure from the transition function s: $S(p, a) = 0.5|p/X, R > +0.5|q/X, R >$, it will get the current state $p$ by calling $NextWord(s,',')$, get the input alphabet $a$ by calling $NextWord(s,')')$. Then, splitting $s$ by '+', it will get two substrings of $0.5|p/X, R >$ and $0.5|q/X, R >$ respectively. For each substring, it can get the transition probability 0.5 by calling $NextWord(s,'|')$, get the next state $p$ or $q$ by calling $NextWord(s,'/')$, and get the move direction L or R by calling $NextWord(s,'>')$ respectively. After interpreting each line of the transition function, a transition table is automatically created as shown in Figure 3:

**Superposition.** Another important contribution of our work is the numerical simulation of the superposition states in QTM. Mathematically, we simulate the transition process from the current measured active state $q_i \in Q$ to the next superposition states (as specified in Equation 3) by randomly choosing and activating one of the multiple states through a random generator $\zeta$ that is varying along with time $t$ s.t. $\zeta(t) \in [0, 1]$. In this way, the mathematical representation of the next active state is calculated as:

$$\delta(q_i, a) = \left\{ q_j/b, X \in \{L, R\} \middle| \zeta(t) \geq \sum_{j=1}^{|Q|} p_{i,j} \right\}, \tag{4}$$

where the transition probability $p_{i,j} \neq 0 \in P$. By setting with different frame refreshing rate $\Delta t$, the frequency of updating the random value $\zeta(t)$ will be adjusted accordingly. It's not hard to prove that when $\lim \Delta t \rightarrow 0$, the flickering active states will closely approximate the superposition of the unmeasured quantum state.

```
type=QTM;                                % Specify the type as QTM
state=p,q,r,f;                           % Specify different states
input=a,b,X,Y,[];                        % Specify input alphabet
init=p;                                  % Specify initial state
final=f;                                 % Specify final state
transit=                                 % Specify transition rules:
{
    S(p,a)=0.5|p/X,R>+0.5|q/X,R>;
    S(q,b)=0.5|q/Y,R>+0.5|r/Y,R>;
    S(r,[])=|f/[],L>;                    % (CurrentState,Read)=
                                            |NextState/Write,Move>
}
test=a,a,a,b,b;                          % Test input symbols
```

**Figure 4: A Quantum Turing Machine (QTM) represented in our proposed C language-like scripts.**

Therefore, our proposed numerical method described in Equation 4 is realistic enough to simulate the superposition states in QTM . Here we show an example from Figure 4 when simulating the transition function of $S(p, a) = 0.5|p/X, R\rangle + 0.5|q/X, R\rangle$ from the current measured active state $p$ to the next superposition states $0.5|p\rangle + 0.5|q\rangle$, the next active state will be either $p$ or $q$ depending on whether the random generator can generate a number less or greater than 0.5. As the chances for both cases are the same , there will be an equal chance to switch to the next active state as $p$ or $q$, this satisfies the definition of the superposition representation.

## 4 EXPERIMENT RESULTS


**Figure 6: Vuforia Marker.**

We developed our AR Quantum Turing Machine (QTM) simulator with one of the most popular development tool for virtual reality applications which is called Unity3D [13] which has integrated APIs which game developers can easily have access to after installing the corresponding SDKs. We construct the AR environment on Vuforia engine [7], which is a comprehensive, scalable enterprise AR platform. Vuforia's wide-ranging solution suites provide practical AR technology that is widely used among researchers, for testing our AR QTM simulator. In our experiment, As shown in Figure 6, the virtual QTM is automatically loaded onto an image marker labeled with "Vuforia" displayed on a screen of a cellphone. As one of the most important features supported by the Vuforia engine, AR content can be directly loaded through a webcam.In our experiment, we simulate three types of quantum automata including QFA, QPDA, and QTM.

**QFA**. Figure 5 shows the experiment results for simulating different types of quantum automata using our proposed AR simulator. The left column in this figure shows the C-like script, the middle column shows the initialization of the simulator (Run in Unity 3D Editor) and the right column shows the simulation result of the AR simulator on webcam based on the Vuforia engine. Different rows demonstrate different types of automata or Turing machines respectively given to different input strings specified in their corresponding

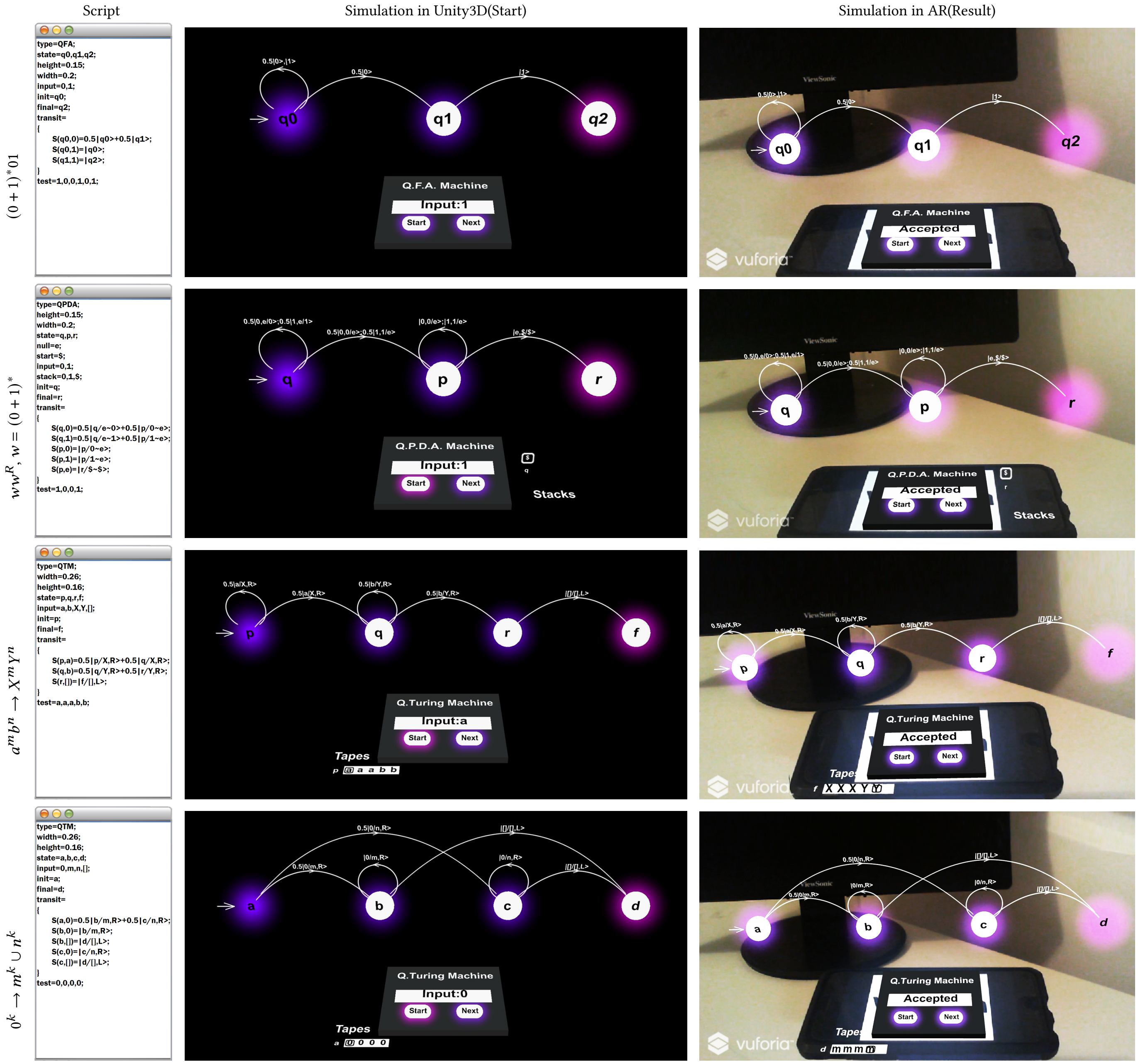
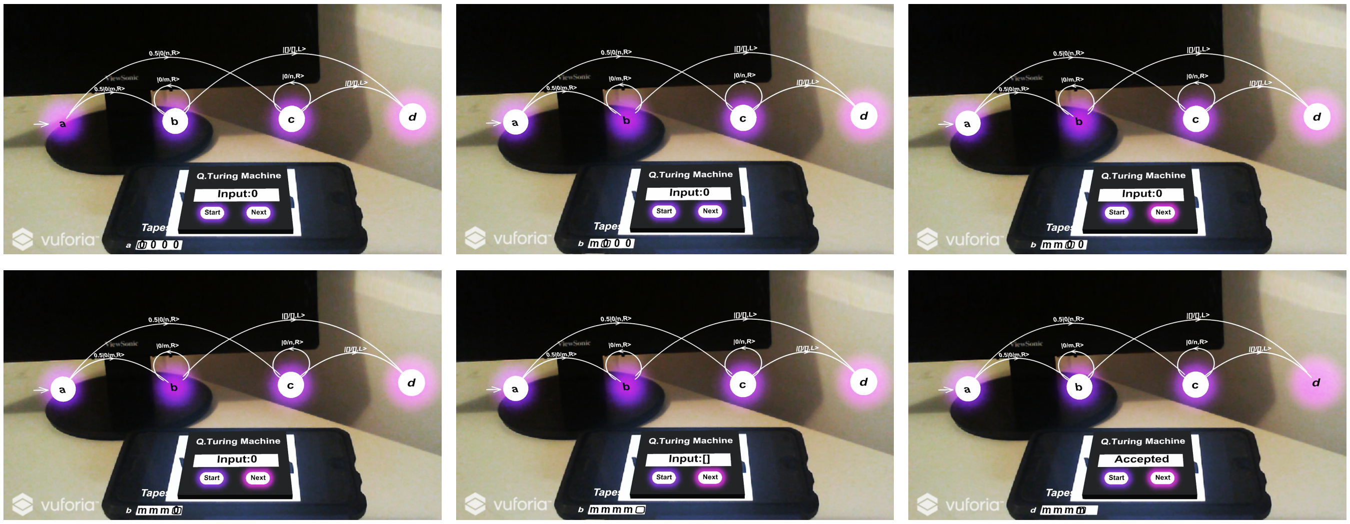


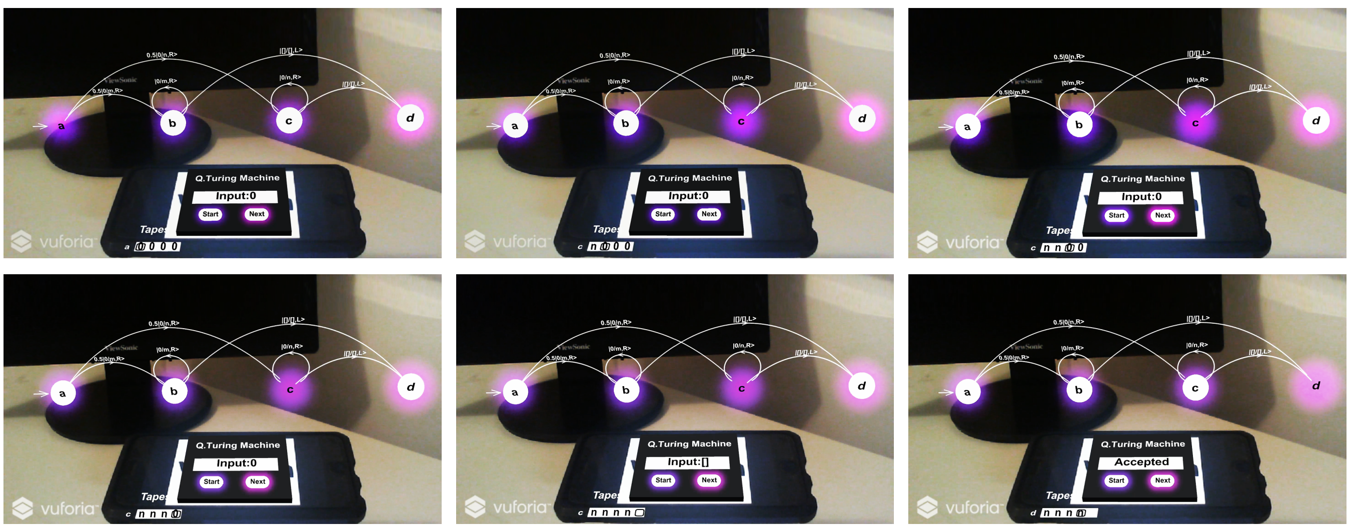

**Figure 5: Experiment Results. This figure shows the experiment results for simulating quantum automata.**

C-like script. The first row shows an example of Quantum Finite Automaton (QFA) that is able to accept 0-1 strings that end with a "01" substring, or, represented by a regular expression: $(0 + 1)^*01$. As we can see from the result on the right column, this simulator reaches a "accept" state, which is highlighted as a pink transparent sphere in the Vuforia AR camera, while taking the input string of "100101" which ends with "01". This result satisfies the definition of this QFA and validates our approach in simulating QFAs.

**QPDA**. The second row in Figure 5 shows the experiment results for simulating a Quantum Pushdown Automaton (QPDA) that accepts 0-1 strings who are palindromes or, represented by a regular expression $ww^R$, $w = (0+1)^*$. Different from the QFA, this example of QPDA is extended with a stack. As we can see from the result on the right column, this simulator reaches a "accept" state with an empty stack while taking the input string of "1001" where w="01" and it is palindromes. This result satisfies the definition of this QPDA and validates our approach in simulating QPDAs.




(a) A case converts "0000" to "mmmm".




(b) A case converts "0000" to "nnnn".

**Figure 7: Simulation process. This figure shows QTM simulation process on different runs with different outputs.**

**QTM**. The third row in Figure 5 shows the experiment results for simulating a Quantum Turing Machine (QTM) that accepts $a^m b^n, m, n = 1, 2, 3, \ldots$ and replace $a \longrightarrow X$ and $b \longrightarrow Y$. Different from the QFA, this example of QTM is extended with a tape. As we can see from the result on the right column, this simulator reaches a "accept" state while taking the input string of "aaabb" (as shown in the left column) and output the string of "XXXYY" onto the tape (as shown in the right column). This result satisfies the definition of this QTM and validates our approach in simulating QTM.

**NTM vs. QTM.** In theoretical computer science, a Nondeterministic Turing Machine (NTM) is a theoretical model of computation whose given rules specify more than one possible action when in some given situations. Unlike a Deterministic Turing Machine (DTM), an NTM's next state is not completely determined by its action and the current symbol it sees, rather, it shows nondeterministic behavior that can transit from one active state to multiple states.

Figure 9 shows an example of NTM that is described with our C-like script. This NTM accepts $0^k, k = 1, 2, 3, \ldots$ and replace $0 \longrightarrow m$ or replace $0 \longrightarrow n$. Mathematically, the accepted input and output for this NTM is represented as a string with this grammar: $0^k \longrightarrow m^k \cup n^k$. Indeed, QTM has the power to do a similar task through multiple runs. The fourth row in Figure 5 shows the experiment results for simulating a QTM that accepts $0^k \longrightarrow m^k \cup n^k$. Figure 7 shows the simulation process on this QTM for two different runs on the same input but results in two different outputs. As we can see from the result in Figure 7 (a), this simulation shows the running

```
type=TM;
width=0.26;
height=0.16;
state=a,b,c,d;
input=0,m,n,[];
init=a;
final=d;
transit=
{
    S(a,0)=b/m,->;
    S(a,0)=c/n,->;
    S(b,0)=b/m,->;
    S(b,[])=d/[],<-;
    S(c,0)=c/n,->;
    S(c,[])=d/[],<-;
}
test=0,0,0,0;
```

**Figure 9: NTM.**


**Figure 8: NTM Comparison. This figure shows the process of simulating a NTM that accepts $0^k \rightarrow m^k \cup n^k$ which is done through the work presented by Li et al. [6].**

process of how this QTM reaches an "accept" state while taking the input string of "0000" and outputting the string of "mmmm" onto the tape in the end. Give the same input and start another simulation on the same QTM, Figure 7 (b) shows a simulation process of how this QTM reaches an "accept" state while taking the input string of "0000" and outputting the string of "nnnn" onto the tape in the end. By comparing this output from QTM and the output from an NTM's simulation process as shown in Figure 9 (this NTM simulation is done through the work presented by Li et al. [6]), we prove that our proposed QTM AR simulator is able to simulate an NTM given multiple runs and visually proves the power of quantum computing using this QTM AR simulator.

## 5 CONCLUSION

In this paper, we propose a novel and practical interactive interface for simulating Quantum Turing Machine (QTM) in Augmented Reality (AR) that is able to simulate and visualize the computational process of different types of quantum automata including Quantum Finite Automaton (QFA), Quantum Pushdown Automaton (QPDA), and Quantum Turing Machine (QTM). By writing an easy-to-use C-like script in our proposed interface that describes a QTM, users can interactively simulate such QTM in an immersive augmented reality platform based on the Vuforia AR engine via a webcam. Through a series of numerical experiments, we validate the correctness of our proposed QTM AR simulator in simulating different quantum automata and we show the great potential to apply our QTM AR simulator to quantum computing education through a realistic and interactive visualization interface in augmented reality.

**Future Work.** Given our successful work on simulating QTM in AR, it's promising to extend this interactive approach and this C-like script notation that is used in this work to simulate other variations of Turing machines such as Probabilistic Turing Machine (PTM) which is an NTM that chooses between the available transitions at each point according to some probability distribution, Alternating Turing Machine (ATM) which is an NTM with a rule for accepting computations that generalizes the rules used in the definition of the complexity classes NP and co-NP, Neural Turing machine which is a recurrent neural network model of a Turing machine, etc. We believe our work can be set up as an example for establishing immersive teaching strategies for STEM education in quantum computing courses and related computational theory courses.

## REFERENCES


[1] Marco Carpentieri. 2003. On the simulation of quantum turing machines. *Theoretical computer science* 304, 1-3 (2003), 103–128.
[2] Max Dauchet. 1989. Simulation of Turing machines by a left-linear rewrite rule. In *International Conference on Rewriting Techniques and Applications*. Springer, 109–120.
[3] Andreas Dengel. 2018. Seeking the treasures of theoretical computer science education: Towards educational virtual reality for the visualization of finite state machines. In *2018 IEEE international conference on teaching, assessment, and learning for engineering (TALE)*. IEEE, 1107–1112.
[4] Paul Adrien Maurice Dirac. 1926. On the theory of quantum mechanics. *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character* 112, 762 (1926), 661–677.
[5] Fred C Hennie and Richard Edwin Stearns. 1966. Two-tape simulation of multitape Turing machines. *Journal of the ACM (JACM)* 13, 4 (1966), 533–546.
[6] Wanwan Li. 2022. Simulating Turing Machine in Augmented Reality : *Full/Regular Research Paper submission for the symposium CSCI-RTED: Education. In *2022 International Conference on Computational Science and Computational Intelligence (CSCI)*. 1–8.
[7] Xinqi Liu11, Young-Ho Sohn, and Dong-Won Park. 2018. Application development with augmented reality technique using Unity 3D and Vuforia. *International Journal of Applied Engineering Research* 13, 21 (2018), 15068–15071.
[8] Quanta Magazine. [n. d.]. *Quantum Computers, Explained With Quantum Physics*. https://www.youtube.com/watch?v=jHoEjvuPoB8
[9] Gerard O'Regan. 2018. Von Neumann Architecture. In *The Innovation in Computing Companion*. Springer, 257–259.
[10] JM Robson. 1984. Fast probabilistic RAM simulation of single tape turing machine computations. *Information and control* 63, 1-2 (1984), 67–87.
[11] Alan Mathison Turing. 1936. On computable numbers, with an application to the Entscheidungsproblem. *J. of Math* 58, 345-363 (1936), 5.
[12] Kamil'A Valiev. 2005. Quantum computers and quantum computations. *Physics-Uspekhi* 48, 1 (2005), 1.
[13] Jingming Xie. 2012. Research on key technologies base Unity3D game engine. In *2012 7th international conference on computer science & education (ICCSE)*. IEEE, 695–699.
[14] Alexander Zable, Lloyd Hollenberg, Eduardo Velloso, and Jorge Goncalves. 2020. Investigating Immersive Virtual Reality as an Educational Tool for Quantum Computing. In *26th ACM Symposium on Virtual Reality Software and Technology*. 1–11.
[15] Yilin Zhu, Feng Pan, Lingxi Li, Xuemei Ren, and Qi Gao. 2015. Simulation of turing machine with uEAC-computable functions. *Mathematical Problems in Engineering* 2015 (2015).