QUANTUM COMPUTER GRAPHICS ALGORITHMS

BY

SIMONA CARAIMAN*

“Gheorghe Asachi” Technical University of Iaşi,
Faculty of Automatic Control and Computer Engineering

Received: September 1, 2012
Accepted for publication: December 17, 2012

Abstract. In this paper we outline the development of a new paradigm in the field of quantum computing: quantum computer graphics. The purpose is to demonstrate how fundamental computer graphics problems can be expressed using the quantum formalism. We show that corresponding quantum computer graphics algorithms can be formulated in order to exploit the immense potential of quantum information processing given by its remarkable properties: inherent parallelism of quantum superpositions, quantum interference and entanglement of quantum states. We introduce quantum solutions for the polygon visibility and global illumination problems and develop the appropriate quantum algorithms.

Key words: quantum computer, quantum information processing, computer graphics, rendering.

2010 Mathematics Subject Classification: 81P68, 68Q05.

1. Introduction

The evolution of current technology allows us to predict that in the following twenty years, the basic element of binary information - the bit - could be implemented at subatomic scale. For example, the $-1/2$ particle spin could be used, or the vertical or horizontal particle polarization, or even the energy level

*e-mail: sarustei@cs.tuiasi.ro
of a hydrogen electron: all these over various possibilities to encode 0 and 1 basis states. At this level, the laws of classical physics cease to apply, while the laws of quantum physics take place, even though the latter ones are not very intuitive. In order to achieve $10^{16}$ gate densities on a chip and frequencies perhaps higher than $10^{15}$ Hz - performances foreseen for 2020 in the case of a “normal” evolution - we have to adopt theoretical models fundamentally and explicitly based on the laws of quantum physics. Instead of eliminating the quantum effects, we will have to exploit them as they can offer by nature a still unimagined computing power.

At present, quantum computation may seem unachievable, but small quantum computers have already been built and larger machines are being developed as described by (Lloyd, 1993; Vandersypen et al., 2000; Ohlsson et al., 2002; Fei et al., 2002; Longdell et al., 2004; Neven et al. 2008; Politi et al., 2009; DiCarlo et al., 2009). These efforts are fueled by an important property: while conventional computers use a binary representation of information that allows the computing power to scale at most linearly with respect to the used resources, quantum computing exploits quantum phenomena which can interact in such a way allowing for an exponential computing power with respect to the number of quantum bits used by the system.

Recent research in the field of quantum computing has proved this immense potential of quantum computing systems to solve problems that are traditionally considered unsolvable due to the necessary computing effort. For example, protein folding illustrates the inherent weakness of the von Neumann computation. Nevertheless, nature solves this problem in a matter of seconds. The “abnormal” efficiency of other biological optimization processes could provide indirect proof of quantum processing when no classical explanation can be foreseen. In this context, even though quantum computers haven’t been built outside research labs, the interest of the scientific community in providing solutions for specific problems in quantum computing is continuously growing.

The remarkable properties of quantum systems led to the emergence of innovative ideas in all major fields of computing, including graphics processing. Basic solutions have been devised for some fundamental computer graphics problems. These solutions are based on expressing these problems in a new manner that allows the use of the quantum formalism (Caraiman & Manta, 2009). Lanzagorta and Uhlmann suggested that quantum algorithms can be devised for the rendering problem (Lanzagorta & Uhlmann, 2005a): Z-Buffering, ray tracing, radiosity and level of detail. Quantum algorithms have also been outlined for some computational geometry problems (Lanzagorta & Uhlmann, 2004) while another important result is the quantum RANSAC algorithm (Caraiman & Manta, 2009). This is an algorithm for robust model fitting in the presence of outliers, with significant applications in both computer graphics and vision. The basic principle used for the quantization of these algorithms consists in reformulating them in order to exploit the special
properties of quantum superpositions of states. In this paper we extend these ideas and provide the quantum algorithms for the polygon visibility problem and global illumination using photon mapping. We show how such an approach allows for these fundamental problems to be solved more efficiently using quantum computation.

The paper is structured as follows. Section 2 introduces the basic principles of quantum computing emphasizing on how information is represented, quantum logical gates and properties of quantum computation that allow for the formulation of powerful quantum algorithms. In Section 3 we briefly overview the main quantum techniques employed in developing the proposed algorithms. In Section 4 we discuss the polygon visibility problem and provide a quantum Z-Buffering algorithm. Section 5 outlines quantum algorithms for global illumination using two well-known techniques: ray tracing and photon mapping. The paper concludes with a discussion over the already accomplished research and the opportunities offered by this emerging field of quantum computer graphics.

2. Basic Concepts in Quantum Computing

The research in quantum informatics appeared as a consequence of Richard Feynman's ideas, winner of a Nobel Prize in physics, who in 1982 suggested that quantum phenomena can be used in conceiving quantum computers. The application of the principles of quantum physics in the computer area led to the concept of quantum computer, in which the data isn't stored in bits like in the conventional memory, but as a combined state of several systems with 2 qubit states.

Just like a classical computer, a quantum computer is built out of three main parts: processor, memory and input/output. A quantum computer can be formally described by 

\[ M = (H, O, T, \delta, \beta), \]

where \( H \) represents the states space (Hilbert space \( \mathbb{C}^{2^n} \)) of the quantum system, \( O \) – the set of unitary transformations, \( T \) – the set of measurement commands, \( \delta \) – is an initialization operator and \( \beta \) – describes the initial measurement. The quantum analogous of the classical bit is the qubit. A qubit is a quantum system whose states can be completely described by the superposition of two orthonormal basis states, labeled \( |0\rangle \) and \( |1\rangle \) (in a Hilbert space \( H = \mathbb{C}^2 \), \( |0\rangle = (1 \ 0)^T \), \( |1\rangle = (0 \ 1)^T \)). Any state \( |\psi\rangle \) can be described by:

\[ |\psi\rangle = \alpha|0\rangle + \beta|1\rangle, \quad |\alpha|^2 + |\beta|^2 = 1, \]  

(1)
where $\alpha$ and $\beta$ are complex numbers. Thus, unlike the classical bit, the qubit can also be in a state different from $|0\rangle$ and $|1\rangle$: linear combinations of states can be formed, called superpositions (1). When measuring a qubit either the result 0 is obtained, with probability $|\alpha|^2$, or 1 with probability $|\beta|^2$. The sum of the probabilities must be 1, so the state of a qubit represents a unit vector in a complex bi-dimensional vector space.

A collection of $n$ qubits is called a quantum register with dimension $n$. The general state of an $n$-qubit register is

$$|\psi\rangle = \sum_{i=0}^{2^n-1} a_i |i\rangle,$$

where $a_i \in C$, $\sum_{i=0}^{2^n-1} |a_i|^2 = 1$. This means that the state of an $n$-qubit register is represented by a complex unit vector in Hilbert space $H_{2^n}$.

2.1. Reversibility of Quantum Computation

To achieve a coherent computation, the quantum registers need to be isolated such that there is no interference with the environment. The entropy of such a system needs to remain constant because heat dissipation is not allowed. Thus, a change in the state of the system should be produced adiabatically which implies that all computation processes have to be reversible. Any reversible operation can be described by a unitary operator, $U$, with $U^{-1} = U^*$. The composition of unitary operators is also unitary because $(UV)^{-1} = V^*U^*$.

A general unitary transformation in the bi-dimensional Hilbert space $C^2$ can be defined by:

$$U = \begin{pmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \end{pmatrix}.$$

2.2. Measurement of Quantum States

Measuring the state of $n$ qubits reduces the dimensionality of the $H$ space with a factor of $2^n$. The result of the measurement is influenced by the probability amplitude of a certain configuration of bits. Let $|\psi\rangle$ be the state of two quantum register with $n$ and $m$ qubits respectively:
\[ |\psi\rangle = \sum_{i=0}^{2^n-1} \sum_{j=0}^{2^m-1} c_{i,j} |i,j\rangle \text{ with } \sum_{i,j} c^*_{i,j} c_{i,j} = 1. \] (4)

Base vectors \(|i,j\rangle\) are interpreted as a pair of binary numbers with \(i < 2^n\) and \(j < 2^m\). Probability \(p(I)\) of measuring number \(I\) in the first register and the post-measurement state are given by:

\[
p(I) = \sum_{j=0}^{2^m-1} c^*_{I,j} c_{I,j} \text{ and } |\psi_I\rangle = \frac{1}{\sqrt{p(I)}} \sum_{j=0}^{2^m-1} c_{I,j} |I,j\rangle.
\] (5)

Measurement is the only non-unitary operation that a quantum computer must be able to execute during a computation.

### 2.3. Quantum Gates

The quantum analogous of the classical NOT gate is labeled \(X\) and can be defined such that \(X|0\rangle = |1\rangle\) and \(X|1\rangle = |0\rangle\). The quantum NOT gate acts similarly with its classical counterpart, although, unlike in the classical case, its action is linear: state \(\alpha|0\rangle + \beta|1\rangle\) is transformed in a corresponding state \(\beta|0\rangle + \alpha|1\rangle\). A convenient way of representing the action of the quantum NOT gate is in matrix form:

\[
X = \begin{pmatrix}
0 & 1 \\
1 & 0
\end{pmatrix}.
\] (6)

The Hadamard gate, given by the following matrix form

\[
H = \frac{1}{\sqrt{2}} \begin{pmatrix}
1 & 1 \\
1 & -1
\end{pmatrix},
\] (7)

is sometimes described as “square root of NOT” because when applying it to any of the \(|0\rangle\) or \(|1\rangle\) basis states produces an equal mixture of both of them:

\[
H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \text{ and } H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle).
\] (8)
Controlled gates are quantum logical gates acting on more than one qubit. The notion of controlled gate allows the implementation of if-else constructs. Quantum controlled gates use a control qubit to determine whether a specific unitary action is applied to a target qubit.

The controlled-NOT operator (CNOT) is the prototypical multi-qubit gate. The first parameter of a CNOT gate is the control qubit. If this qubit is in state $|0\rangle$, the target qubit is left unchanged and if the control qubit is in state $|1\rangle$, the target qubit is flipped:

$$
\begin{align*}
|00\rangle &\rightarrow |00\rangle; \\
|01\rangle &\rightarrow |01\rangle; \\
|10\rangle &\rightarrow |11\rangle; \\
|11\rangle &\rightarrow |10\rangle.
\end{align*}
$$

The CNOT operator is a generalization of the classical XOR, since its action can be summarized as $[x, y] \rightarrow [x, x \oplus y]$, where $\oplus$ is addition modulo two. The matrix representation of CNOT is:

$$
CNOT = \begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0
\end{pmatrix}. \tag{9}
$$

There are several other multi-qubit gates. Nevertheless, the controlled-NOT gate and the single qubit gates represent the prototypes for any other quantum gate because of the following remarkable universality result: any multi-qubit gate can be built out of CNOT gates and single qubit gates. The proof of this statement represents the quantum analogous to the universality of the classical NAND gate.

2.4. Quantum Algorithms

According to the principles of quantum physics the computing power of a quantum machine is immense compared to the one of a classic computer due to three remarkable quantum resources. *Quantum parallelism* harnesses the superposition principle and the linearity of quantum mechanics in order to compute a function simultaneously on arbitrarily many inputs. *Quantum interference* makes it possible for the logical paths of a computation to interfere in a constructive or destructive manner. As a result of interference, computational paths leading to desired results can reinforce one another, whereas other computational paths that would yield an undesired result cancel each other out. Finally, there exist multi-particle quantum states that cannot be described by an independent state for each particle. The correlations offered by
these states cannot be reproduced classically and constitute an essential resource of quantum information processing called *entanglement*.

These remarkable properties of quantum systems allowed the formulation of optimal algorithms for two fundamental problems: integer factorization (Shor's algorithm – (Shor, 1994)) and the search in an unstructured database (Grover's algorithm – (Grover, 1996)). Thus, two main classes of quantum algorithms have better time complexity than their classical counterparts. The first one is based on the quantum Fourier transform (detailed by (Nielsen & Chuang, 2000)) and includes remarkable algorithms for solving the factorization and discrete logarithm problems, while providing exponential speedup over the best known classical algorithms. The second class of algorithms is based on the quantum search algorithm described by (Grover, 1996). This class of algorithms provides a quadratic speedup with respect to the best classical algorithms. The importance of the quantum search algorithm is underlined by the intensive use of search techniques in classical algorithms, while, in many cases, a natural adaptation is possible in order to provide a faster quantum algorithm as proved by (Caraiman & Manta, 2009). Special interest is given to the quantum counting algorithm presented by (Brassard et al., 2000) which represents an inspired combination of quantum search with the Fourier transform and allows the estimation of the number of solutions to a search problem faster than possible on a classical computer.

The main source of computational power of quantum systems is represented by the inherent parallelism of quantum computation. Quantum parallelism is a fundamental feature for many quantum algorithms. In a simplified manner, this parallelism allows quantum computers to evaluate a function, \( f(x) \), for many values of \( x \) simultaneously, by creating superposition states.

Nevertheless, this parallelism isn't immediately useful because measuring a superposition state (containing all values \( f(x) \)) has the effect of collapsing the superposition in one of the basis states, thus only giving the value of the function for one single value for \( x \). Quantum computing needs more than parallelism in order to be useful: the ability to extract information about more than one value of \( f(x) \) from superposition states. This can be realized by combining quantum parallelism with the interference of quantum states and it is exploited by the known quantum algorithms that perform better than the classical counter-parts.

These results determined not only the investigation of innovative applications that can be developed using these high performance computing systems, but also of ways to improve the performances over the classical case. Exploiting the properties of quantum systems and these fundamental results recently led to the emergence of innovative ideas in the field of computer graphics.

Many problems in classical computing can be reformulated to express the search of a unique element that satisfies a certain predefined condition. If there is no additional information about the search condition, the best classical algorithm is a brute-force search, meaning that the elements are sequentially tested against the search condition. For a list of $N$ elements, this algorithm executes an average of $N/2$ comparisons. By exploiting the advantages of quantum parallelism and interference of quantum states, (Grover, 1996) formulated a quantum algorithm that can find the searched element in an unstructured database in only $O(\sqrt{N})$ steps.

Other recent results allow quantum algorithms to be used in solving search problems involving multiple solutions. These results include the semicloning protocol and the algorithm for the extraction of a semiclon state (Lanzagorta & Uhlmann, 2005b). They enlarge the spectrum of problems that can be efficiently solved using quantum parallelism.

Grover’s results are only valid for the case in which the number of elements satisfying the search condition is $t = 1$. Nevertheless, in most practical applications $t$ is unknown and the extraction of the entire set of $t$ solutions is required. There are variations of Grover’s algorithm that can randomly extract one of the solutions when $t$ is unknown by applying $\left\lfloor \frac{\pi}{4} \frac{\sqrt{N}}{t} \right\rfloor$ iterations for $t < \frac{N}{2}$ as in (Boyer et al., 1996). Extracting the entire set of solutions requires the determination of number $t$ prior to executing the $O\left( \frac{N}{\sqrt{t}} \right)$ iterations of Grover’s algorithm. The optimal quantum algorithm which determines the number $t$ of solutions requires $O\left( \sqrt{t(t+1)(N-t+1)} \right)$ time (the approximate counting algorithm described by (Brassard et al., 2000)) and dominates the complexity of Grover’s algorithm for the case when $t$ is unknown. Nevertheless, knowing $t$ doesn’t allow for the sequential extraction of the solutions because each time a state is measured, the solution superposition is destroyed. Thus the repeated application of Grover’s algorithm would be needed in order to randomly extract the solutions and the number of searches applied for determining all $t$ solutions is $O(t \log t)$. This would imply a total complexity of $O(tN \log t)$. The semicloning protocol (Lanzagorta & Uhlmann, 2005b) eliminates the necessity to randomly extract the solutions, thus obtaining an optimal complexity, $O(t)$. 
Another result with significant importance to quantum search is the quantum algorithm for finding the minimum/maximum value in an unstructured data set. This can be done in $O(\sqrt{N})$ steps (Durr & Hoyer, 1999).

In the following we will show how these algorithms can be exploited in order to devise quantum solutions for several fundamental computer graphics problems.

In computer graphics each object that needs to be visualized is decomposed in thousands or millions of polygons or other such surfaces. These surfaces are stored in a database, which for most practical applications is very large, for example several million elements, depending on the scene complexity. In order to render such a scene, the visualization system applies a variety of rendering algorithms to each element of the database. These rendering algorithms have a common characteristic of executing several searches in the scene database in order to find which ones are visible. Taking into account the large number of objects in the database, this search process tends to represent the most significant source for latency. Thus, these search operations are primary candidates for optimization using Grover's quantum search algorithm or any of its variants. Using such an approach we show how quantum algorithms can be devised for determining the polygon visibility using the Z-Buffering algorithm and for global illumination models such as photon mapping.

### 4. The Polygon Visibility Problem

In order to render a 3D scene it is necessary to determine the visibility of the geometric primitives that form the objects of the scene. One of the simplest methods that can be used is the Z-Buffering algorithm originally proposed by (Catmull, 1975).

A Z-Buffer has the same dimensions as the color buffer and for each pixel stores the depth value $z$ of the closest geometric primitive, i.e. the distance between the camera position and the respective primitive. When processing a primitive, the associated $z$ value is computed and compared to the content of the Z-Buffer in the current pixel. If this new value is smaller than the one stored in the Z-Buffer, the color of the current pixel and the Z-Buffer are updated according to the primitive being processed. Otherwise, the color buffer and the Z-Buffer are left unchanged and the processing continues with the next primitive (Algorithm 1).

Although simple, this algorithm is also very efficient. In order to apply this algorithm it is necessary to scan-convert each polygon in the scene. Thus, each polygon is projected onto the screen and the pixels corresponding to this projection are determined. Then, for each such pixel, the distance to the polygon is computed.
The algorithm iterates for each polygon in the scene and the entire operation needs $O(pN)$ time, where $p$ is the average number of projection pixels for a polygon. In other words, the problem can be solved in $O(p)$ time for each polygon. In terms of depth complexity of the scene, $O(Pd)$ steps are needed, where $P$ is the number of pixels in the image and $d$ is the average number of polygons projected on the same pixel. Determining the pixels intersected by a certain polygon can be done using a scan-line algorithm. Thus, polygon rasterization using the Z-Buffering algorithm has $O(N(P + p))$ time complexity.

**Algorithm 1** The Z-Buffering Algorithm

1: /* input - database containing the $N$ polygons in the scene */
2: /* output - $FB$, color buffer of the resulting image and $ZB$ - depth buffer associated with the resulting image */
3: initialize $ZB$
4: for all polygons $p_i$ in the scene do
5:    determine the pixels that intersect $p_i$ (scan-line algorithm)
6:    for all pixels $(x, y)$ that intersect $p_i$ do
7:       compute $z_{p_i}(x, y)$
8:       if $z_{p_i}(x, y) < ZB(x, y)$ then
9:          $ZB(x, y) = z_{p_i}(x, y)$
10:         $FB(x, y) = I_{p_i}(x, y)$
11:    end if
12: end for
13: end for

Even though the Z-Buffering algorithm is not necessarily a search problem, a quantum approach for this algorithm can be built by exploiting a variant of Grover's algorithm: the quantum algorithm for searching the minimum value in a data set. The main idea behind this approach was suggested by (Lanzagorta & Uhlmann, 2005a) and based on it we provide a quantum algorithm for determining the visibility of geometric primitives in a 3D scene (Algorithm 2).

For each pixel in the resulting image the quantum Z-Buffering algorithm uses a superposition of quantum states where each element represents one of the $N$ polygons of the scene. In order to determine the polygons that intersect the current pixel we can apply the quantum search algorithm with multiple solutions. As discussed in the beginning of this section, this stage is
executed in $O\left(\sqrt{dN}\right)$, where $d$ represents the number of polygons satisfying the search condition.

The distance $z_p$, between the current pixel and polygon $i$ can be computed for all polygons $p_i$ that intersect the current pixel. Exploiting the natural parallelism of the quantum computation allows for this step to be executed in $O(1)$ time. Out of these distances we have to find the shortest one and use the corresponding polygon to color the current pixel. This can be done in $O(\sqrt{d})$ time using the quantum min search algorithm.

**Algorithm 2** Quantum algorithm for determining the visibility of geometric primitives in a 3D scene

1: /* input - database containing $N$ polygons */
2: /* output - $FB$, color buffer of the resulting image */
3: for all pixels $(x, y)$ in the image do
4: create a uniform superposition of states, $|\psi_0\rangle$, where each element represents one of the $N$ polygons in the scene, $\{P_1, P_2, \ldots, P_N\}$,
   $$|\psi_0\rangle = \frac{1}{\sqrt{N}}(|P_1\rangle + |P_2\rangle + \ldots + |P_N\rangle)$$
5: apply the approximate counting algorithm described by (Brassard *et al*., 2000), where a solution state corresponds to polygons intersecting pixel $(x, y)$
6: apply the algorithm described by (Lanzagorta & Uhlmann, 2005b) to extract the $d$ solutions of the search problem in step 5
7: create a uniform superposition of states, $|\psi_1\rangle$, where each element represents one of the $d$ polygons that intersect the current pixel, $\{P_1, P_2, \ldots, P_N\}$,
   $$|\psi_1\rangle = \frac{1}{\sqrt{d}}(|P_1\rangle + |P_2\rangle + \ldots + |P_d\rangle)$$
8: $|\psi_2\rangle = \text{determineZ}(|\psi_1\rangle)$
9: $P_m = \text{detMin}(|\psi_2\rangle)$ /* determine the minimum value using the algorithm described by (Durr & Hoyer, 1999) */
10: $FB(x, y) = I_{P_m}(x, y)$
11: end for
The complexity of the quantum Z-Buffering algorithm is $O(P\sqrt{dN})$, while the classical algorithm needs $O(P(d + N))$ steps. Here $P$ represents the total number of pixels, $d$ is the average number of polygons intersecting one pixel and $N$ is the number of polygons in the scene. The main advantages of the quantum variant of the algorithm are the fact that it has a much better scalability with respect to the number $N$ of polygons than its classical counterpart and that it copes with arbitrarily shaped objects and it is not limited to simple polygons.

5. Global Illumination

In Computer Graphics, the rendering equation describes the flow of light energy in a 3D scene. Based on the physics of light, it provides theoretically perfect results, in contrast to the various rendering techniques, which approximate this ideal.

The physical basis for the rendering equation is the law of energy conservation. At a particular position and direction, the outgoing light ($L_o$) is the sum of the emitted light ($L_e$) and the reflected light ($L_r$). The reflected light itself is the sum of the incoming light ($L_i$) from all directions, multiplied by the surface reflectivity and incoming angle (Dutre et al., 2006):

$$L_o(x, y) = L_e(x, y) + \int_{\Omega} f_r(x, w', w) L_i(x, w') (w' \cdot n) dw'$$

where: $L_o(x, w)$ is light outward at a particular position $x$ and in direction $w$, $L_e(x, w)$ – light emitted from the same position and direction, $\int_{\Omega} \cdots dw'$ – an infinitesimal sum over a hemisphere of inward directions, $f_r(x, w', w)$ – the proportion of light reflected at the position $x$ (from inward direction to outward direction), $L_i(x, w')$ – light inward from the position $x$ and direction $w'$, $(w' \cdot n)$ – the attenuation of inward light due to incident angle.

Solving the rendering equation for any given scene represents the main challenge in realistic rendering. One approach for solving this equation is based on finite element methods, leading to the radiosity algorithm. Another approach using Monte Carlo methods has led to many different algorithms including ray tracing and photon mapping.

In the following we present a quantum implementation for the global illumination problem based on photon mapping. We first analyze the ray tracing
technique and provide the corresponding quantum algorithm as it integrates well in the first step of photon mapping.

5.1. Ray Tracing

Ray tracing, originally described by Whitted (1980), is one of the most used rendering techniques. One such algorithm determines the visibility of the surfaces by tracing a ray from the observer to the objects in the scene. The intersection between a ray and on object determines the color of the pixel. This process takes place for each pixel on the screen. Thus, for each pixel on the screen, the algorithm determines whether there is an intersection between the ray and any of the objects in the scene. If the ray intersects more than one object, only the closest one from the screen is displayed. Thus, the classical algorithm executes $O(N)$ operations per ray, because it determines the intersections for each of the N objects in the scene.

Ray tracing is by nature a search algorithm, thus it is a good candidate for optimization using Grover's quantum search algorithm (Lanzagorta & Uhlmann, 2005a). In this way ray tracing can be implemented by executing a quantum search for the intersections between rays and polygons, followed by a quantum search for the polygon closest to the screen.

For a quantum implementation of the ray tracing algorithm, a quantum state $|\psi\rangle$ is created encoding all polygons in the scene in a uniform superposition:

$$|\psi\rangle = \alpha (|00\ldots01\rangle + |00\ldots10\rangle + \ldots).$$ (11)

This superposition of states is used for each traced ray. It is also necessary to define a function $f$ that would act on each state of the superposition like in the following:

$$f(x) = \begin{cases} 1, & x \text{ intersects the ray} \\ 0, & \text{otherwise} \end{cases}. \quad (12)$$

Quantum parallelism can be employed in order to evaluate $f(x)$ for each element of the superposition (each object in the scene) in a single computational step. Next, the semicloning protocol can be used to determine all $k$ objects intersecting the ray in $O(\sqrt{Nk})$ time. The object closest to the screen can now be determined using the quantum algorithm for determining the minimum in $O(\sqrt{k})$ steps. The total complexity of the quantum ray tracer becomes $O(\sqrt{Nk} + \sqrt{k})$ per ray. In most practical application $k \approx O(1)$ and $k \ll N$. 

Algorithm 3 Quantum ray tracing algorithm

1: for all pixels \((x, y)\) in the image do
2: calculate trajectory of ray striking that pixel
3: create a uniform superposition of states, \(|\psi_0\rangle\), where each element represents one of the \(N\) objects in the scene, \(\{O_1, O_2, \ldots, O_N\}\),
\[
|\psi_0\rangle = \frac{1}{\sqrt{N}} (|O_1\rangle + |O_2\rangle + \ldots + |O_N\rangle)
\]
4: apply the approximate counting algorithm described by (Brassard et al., 2000), where a solution state corresponds to objects intersecting the current ray
5: apply the semiclassical algorithm described by (Lanzagorta & Uhlmann, 2005b) to extract the \(k\) solutions of the search problem in step 4
6: create a uniform superposition of states, \(|\psi_1\rangle\), where each element represents one of the \(k\) objects that intersect the current ray, \(\{P_1, P_2, \ldots, P_d\}\),
\[
|\psi_1\rangle = \frac{1}{\sqrt{d}} (|P_1\rangle + |P_2\rangle + \ldots + |P_d\rangle)
\]
7: \(|\psi_2\rangle = \text{computeDistanceFromCrtPixel}(|\psi_1\rangle)\)
8: \(O_m = \text{detMin}\left(|\psi_2\rangle\right) / *\) determine the object with minimum distance from the current pixel using the algorithm described by (Durr & Hoyer, 1999) */
9: \(\text{directIllumColor} = \) calculate contribution of all lights at intersection point
10: \(\text{reflectedColor} = \) recursively trace specularly reflected ray
11: if object is transparent then
12: \(\text{transmittedColor} = \) recursively trace transmitted ray
13: end if
14: \(FB(x, y) = \text{directIllumColor} + \text{reflectedColor} + \text{transmittedColor}\)
15: end for

The main advantage of this quantum solution over the classical case is the optimal time complexity for general object queries, with linear space complexity. In the classical case, polygons are used because computing the intersection is easier than in the case of generic objects.

Ray tracing is an excellent method for simulating reflected and refracted light. Nevertheless, it approximates the diffuse light in a rudimentary and expensive way. The radiosity algorithm is a method that models very well diffuse light, but offers a poor representation of reflected light (Cohen & Wallace, 1993).
5.2. Photon Mapping

Photon mapping is a two-step global illumination algorithm developed by (Jensen, 2001) as an efficient alternative to pure ray tracing Monte Carlo techniques.

Ray tracing systems cannot generate the caustic phenomenon correctly, cannot render indirect illumination produced by objects that reflect light and cannot implement diffuse inter-reflection (thus nor color bleeding). On the other hand, radiosity methods easily produce diffuse inter-reflections and indirect light, but cannot cope with specular reflection, have difficulties in processing transparency, need scene sub-divisioning and are time consuming (a second pass is needed to produce reflection and refraction).

The idea behind photon mapping is to decouple the representation of the scene from its geometry and to store the illumination information in a global spatial data structure called photon map. The first step of the method builds the photon map by tracing photons leaving from a light source, while in the second step the scene is rendered using the information stored in the photon map.

In the first stage, the light sources send packets of light (photons) into the scene. When a photon intersects a surface, the intersection point, incident direction and photon power are stored in the photon map. Typically, two photon maps are created for each scene: one for the caustic effects and one for other types of illumination. After computing the intersection, the surface material provides a probability that the photon is reflected, absorbed or refracted/transmitted. A Monte Carlo method, called the Russian Roulette, is used to choose one of these actions. If the photon is absorbed, its tracing stops. If the photon is reflected or refracted, the surface properties and incident angle are used to compute its new direction.

In the second stage, the photon map is used to estimate the radiosity of each pixel in the final image. For each pixel, a ray is casted into the scene and its first intersection with an object in the scene is computed. The rendering equation is used to compute the radiosity leaving the surface from the intersection point in the direction of the ray. For efficiency, the equation is decomposed into four factors: direct illumination, specular reflection, caustic refraction and indirect illumination.

A quantum implementation of photon mapping can benefit from the special properties of quantum computation under two aspects. On one hand, searching the intersections between a photon and a collection of \( N \) objects in the scene and determining the first intersection can be accelerated using quantum search algorithms just like in the case of the ray tracing algorithm. Thus, the first step of the quantum photon mapping algorithm has \( O(\sqrt{N}) \) time complexity, unlike \( O(N) \) in the classical case.
On the other hand, unlike classical computers, quantum systems can implement Monte Carlo methods using authentic random values. In classical computing, pseudo-random values are used (in fact, fully deterministic numbers). The Monte Carlo techniques are used in photon mapping for two purposes: photon emission by a light source and to probabilistically decide whether at the intersection with a surface the photon is absorbed, reflected or refracted, depending on the surface material (technique called the Russian Roulette).

Using a quantum computer as a random number generator is very simple and assumes applying the Hadamard transform to a quantum state that encodes the probability distribution. The effect of the Hadamard transform is to create uniform superposition of states and thus each state has the same probability to be measured:

\[ H(|0\rangle|0\rangle|0\rangle) = \frac{1}{2^{m/2}} \sum_{i=0}^{2^m-1} |i\rangle \xrightarrow{\text{measure}} |i\rangle \]

with probability \(2^{-m}\)

6. Discussion

Theoretical and experimental research accomplished by now in the field of quantum computing is in an initial stage. Still a lot of effort is needed for clarifying the theoretical and practical aspects concerning the information coding and processing based on quantum phenomena. Nevertheless, it has become an important area of research in computer science and the increased interest is justified by the existence of efficient quantum algorithms that can perform some calculations significantly faster than on classical computers. Building a working quantum computer represents a very challenging task that requires spending a huge amount of resources. Developing efficient quantum algorithms for practical problems would therefore help in justifying these immense efforts. In this paper we have shown how existing quantum algorithms (such as Grover's algorithm and its variants or the quantum Fourier transform) can be used to improve the performance of several fundamental computer graphics algorithms. The speedups of these quantum variants can also be used to increase the realism of the rendered images. Defining the quantum variants for these algorithms was possible by reformulating them in order to exploit the special properties of the superpositions of quantum states. In this way, we identified the steps of these algorithms that could be stated in the terms of a search problem and provided the variants of Grover's algorithm suited for solving these specific steps. The performances of the resulting quantum algorithms are orders of magnitude faster than the classical variants and represent a promising result for the investigation of more such applications of
quantum computing to computer graphics and vision tasks. As far as quantum image processing is concerned, only a few efficient operations are known. Thus, the design of fast algorithms for this field implies extending the knowledge on fundamental and efficient operations.

Acknowledgements. This paper was supported by the project PERFORMERA “Postdoctoral Performance for Integration in the European Research Area” (ID-57649), financed by the European Social Fund and the Romanian Government.

REFERENCES


ALGORITMI CUANTICI PENTRU PROCESARE GRAFICĂ

(Rezumat)

În această lucrare sunt analizate modalitățile prin care proprietățile specifice calculului cuantic pot fi explotaate pentru a formula soluții bazate pe procesarea cuantică a informației pentru probleme specifice grafică. Scopul este de a analiza probleme fundamentale din procesarea grafică în termeni formalismului cuantic. Soluțiile cuantice pentru aceste probleme sunt definite astfel încât să fie posibilă exploatarea imensului potențial de calcul datorat proprietăților remarcabile specifice sistemelor cuantice: paralelismul inerent al superpozițiilor cuantice, interferența cuantică și corelația stărilor cuantice. În lucrarea de față sunt introduse soluții cuantice pentru probleme de randare precum determinarea vizibilității poligoanelor și iluminarea globală. Sunt formuleți și analizați algoritmi cuantici corespunzători.