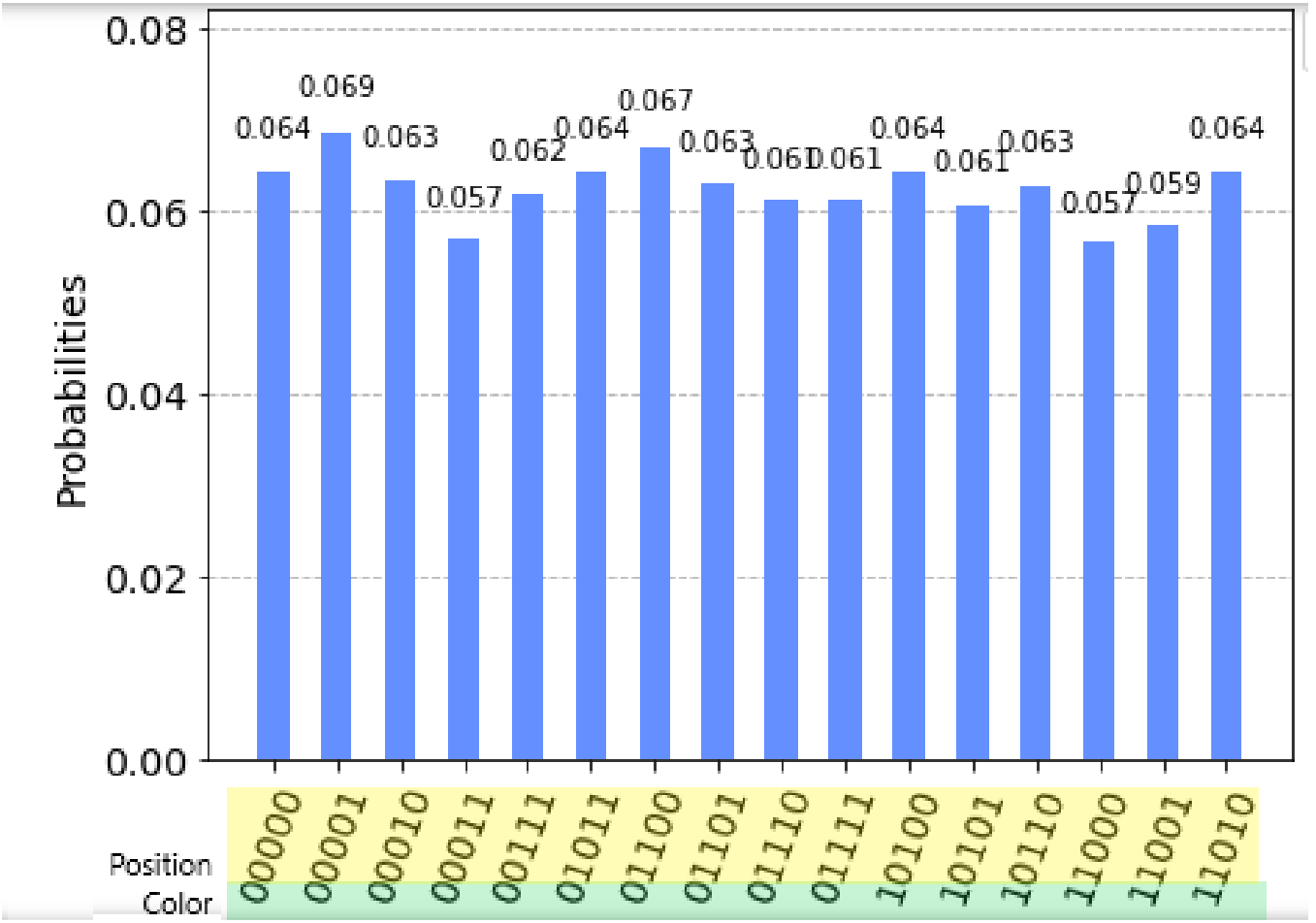
Implementation of FRQI Method using Qiskit

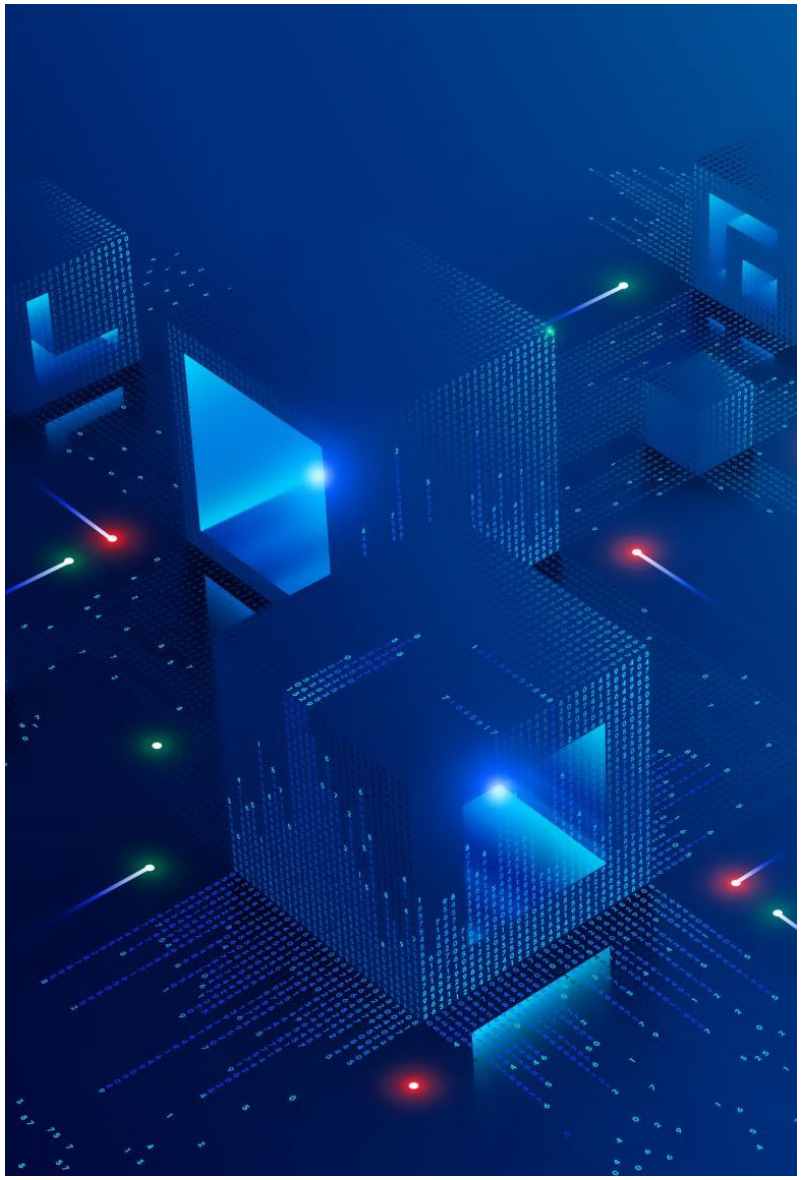
QCourse 570-1

Guillermo “Bill” Gonzalez

University of Texas at San Antonio

and QWorld





Introduction

Quantum Image Processing

Quantum image processing (QImP) promises capabilities and performance improvements over classical methods. These advantages include speed, efficiency, security and reduced storage requirements.

Use Cases

Common image processing operations are edge detection, image enhancement, pattern recognition and image classification to name a few. These are useful for high level functions like facial recognition and object detection in autonomous vehicles.

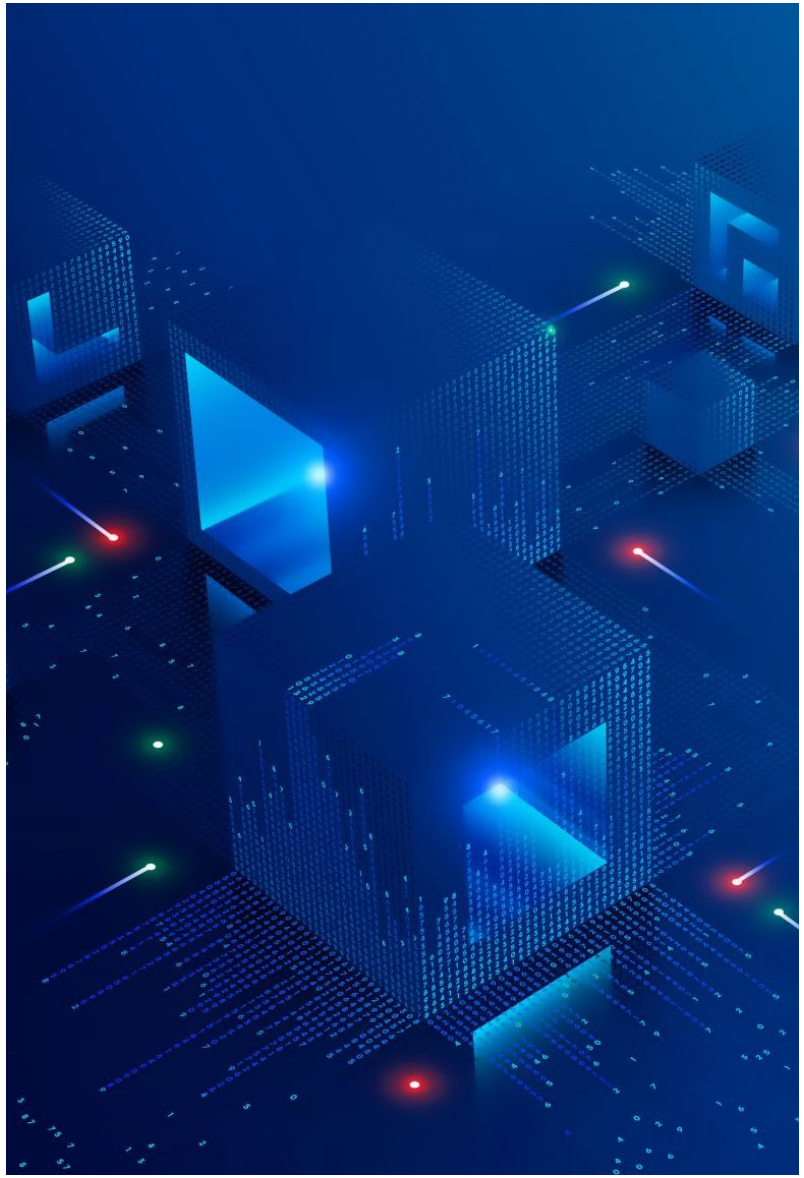
QIR

In order to perform QImP one must first be able to take an image and represent as a set of quantum states. This involves using a method to encode an image into a set of quantum states.

Methods

There are now over thirty different methods for encoding an image into a set of quantum states. Some are variations of the first methods and researchers are constantly hoping to devise more efficient methods that use fewer computing resources.

- Qubit Lattice
- Entangled Image
- Real Ket
- FRQI – flexible representation of quantum image
- NEQR – novel enhanced quantum representation
- NASS – normal arbitrary quantum superposition state
- MCRQI – multi-channel representation of quantum image
- Many, many more…

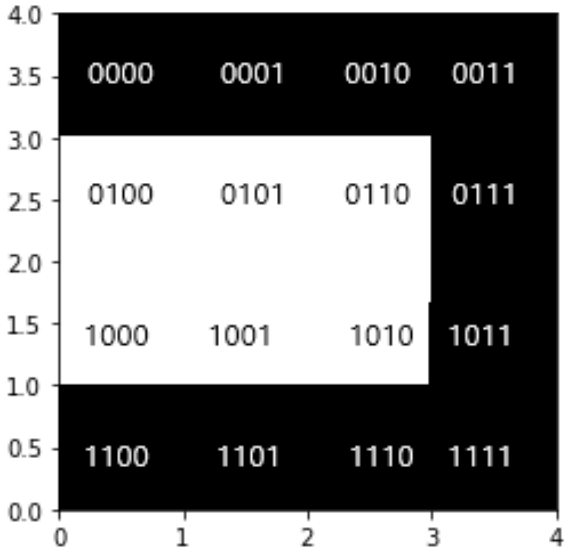


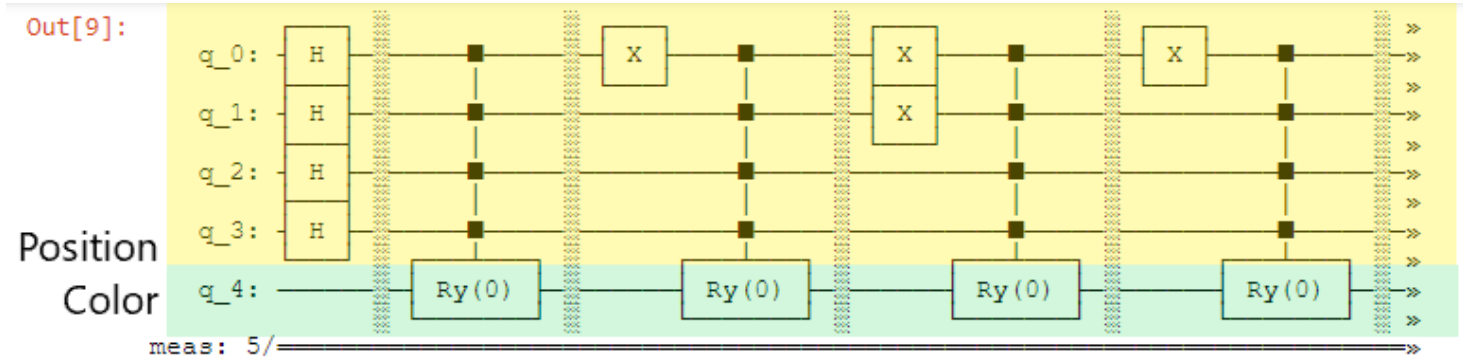
FRQI

One of the most simple and easy to understand encoding methods is Flexible Representation of Quantum Images (FRQI). In this method, an image is broken down into a set of pixels and each pixel is represented as an angle in a qubit. For example, in the image below, the positions of a pixel in a simple 4x4 image are numbered using binary numbers from right to left, top to bottom.

The color of an image is represented as a number from 0 to 255 however for the experiments conducted as part of this project, pixel color values were limited to 0 for black or 1 for white.

```
[[  0   0   0   0]
 [255 255 255   0]
 [255 255 255   0]
 [  0   0   0   0]]
```





Methods

Jupyter Notebooks and Qiskit

After experimenting with various frameworks, all work was performed using an Anaconda environment utilizing a Jupyter Notebook and Qiskit. The working experiment consisted of the following sections

Generic FRQI function with two major functions. This function can take as input an image of any size.

1. Step to keep count of pixel position. Bit mask utilized with logic.
2. Step to encode the “color” portion of the pixel using multi-controlled Ry (rotation) gates

Additional pre-processing functions

1. import a set of simple shapes
2. split data into test and train
3. apply resize
4. greyscale conversion
5. Thresholding

Encoding and Circuit Creation

Lastly, the image was encoded, i.e., the color values were converted to either 0 or 1 and the circuit was built using the generic FRQI function.

Experiments and Results

Image Sizes

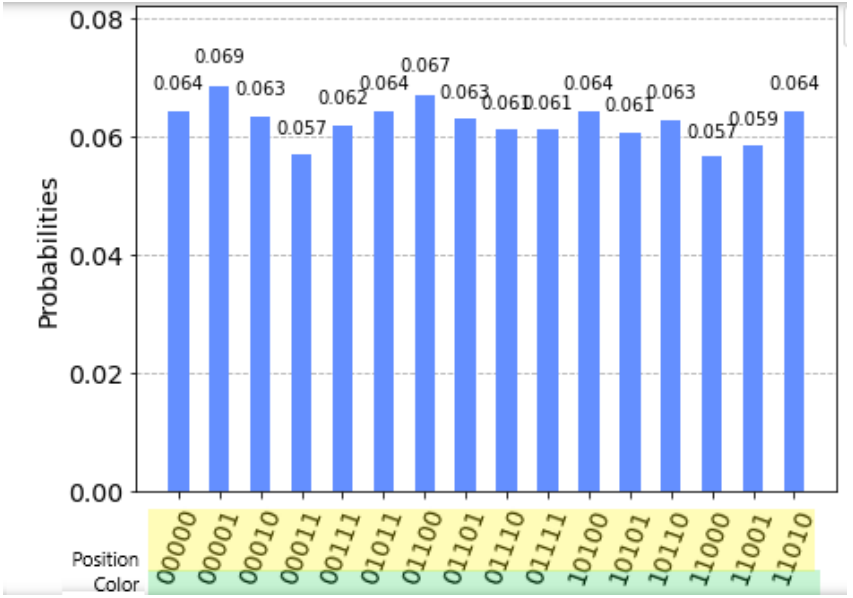
The original image were resized to various sizes like 2x2, 4x4, 8x8 and 32x32. As the sizes of the images were increased, the circuit generation step took considerably longer.

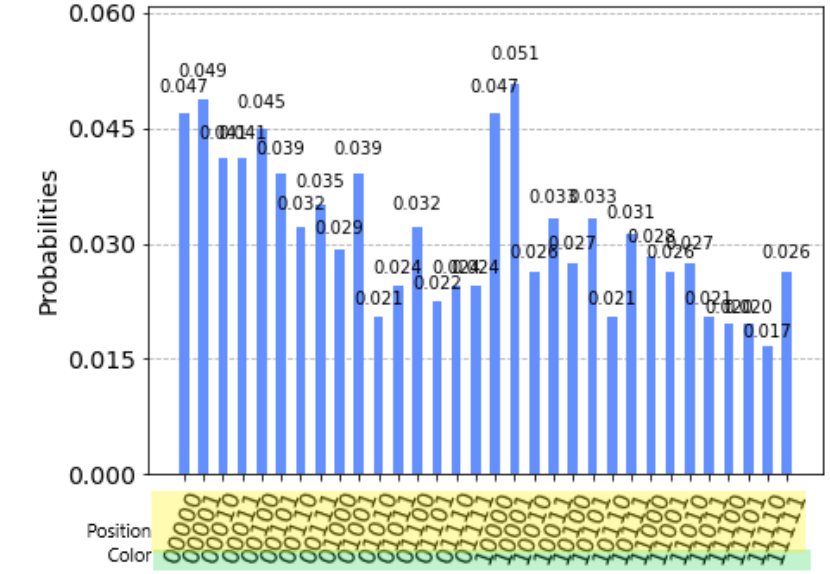
Basic Simulator

The results obtained from running the circuits against a simulator were as expected. The position and number of pixels that were white were identified correctly as were the black pixel positions and count.

Fake Device

Running the circuit on a fake device produced results more likely expected on a real quantum device. All pixel positions were presented as black and white however with some careful analysis, one could determine the correct positions and colors of some of the pixels.





Thank you!