EasyChair Preprint

№ 5356

Quantum Generators: Functional Model for the Self-Assembly of Cells from the Structured Compute Units.

Poondru Prithvinath Reddy

April 19, 2021

Quantum Generators: Functional Model for the Self-Assembly of Cells from the Structured Compute Units.

Poondru Prithvinath Reddy

ABSTRACT

Quantum Generators is a means of achieving mass food production with short production cycles, and when and where required by means of machines rather than land based farming which has serious limitations. The process for agricultural practices for plant growth in different stages is simulated in a machine with a capacity to produce multiple seeds from one seed input using computational models of multiplication (generating multiple copies of kernel in repetition). In this paper, we present a method to produce living natural tissues of crops that are constructed by acquiring several small and more detailed images of large objects and then compose a larger image structure by combining these small images. The main goal of this method is to create structures that are identical to the natural structure that are found in the tissues in the crops. Hence we define functions that support cell types and the composition of cell matrix. We look at Models for creating cell types which will be then self-assembled to create building blocks of natural tissues with independent arrangement and patterning to provide the required architecture. Results show that the proposed method performs well with sample cell images obtained from the Internet.

INTRODUCTION

A **Quantum** (plural quanta) is the minimum amount of any physical entity (physical property) involved in an interaction. On the other hand, **Generators** don't actually create anything instead, they generate quantity prescribed by physical property through multiplication to produce high quality products on a mass scale. The aim of Quantum Generators is to produce multiple seeds from one seed at high seed rate to produce a particular class of food grains from specific class of **seed** on mass scale by means of machine rather than land farming.

The process for agricultural practices include preparation of soil, seed sowing, watering, adding manure and fertilizers, irrigation and harvesting. However, if we create same conditions as soil germination, special watering, fertilizers addition and plant growth in different stages in a machine with a capacity to produce multiple seeds from one seed input using computational models of multiplication(generating multiple copies of kernel in repetition) then we will be closure to achieving mass food production by means of quantum generators(machine generated) rather than traditional land based farming which has very serious limitations such as large space requirements, uncontrolled contaminants, etc. The development of Quantum Generators requires specialized knowledge in many fields including Cell Biology,

Nanotechnology, 3D Cellprinting, Computing, Soil germination and initially they may be big occupying significantly large space and subsequently small enough to be placed on roof-tops.

The Quantum Generators help world meet the food needs of a growing population while simultaneously providing opportunities and revenue streams for farmers. This is crucial in order to grow enough food for growing populations without needing to expand farmland into wetlands, forests, or other important natural ecosystems. The Quantum Generators use significantly less space compared to farmland and also results in increased yield per square foot with short production cycles, reduced cost of cultivation besides easing storage and transportation requirements.

In addition, Quantum Generators Could Eliminate Agricultural Losses arising out of Cyclones, Floods, Insects, Pests, Droughts, Poor Harvest, Soil Contamination, Land Degradation, Wild Animals, Hailstorms, etc.

Quantum generators could be used to produce most important *food* crop *like* rice, wheat and maize on a mass scale and on-demand when and where required.

Computers and Smartphones have become part of our lives and Quantum Generators could also become very much part of our routine due to its potential benefits in enhancing food production and generating food on-demand wherever required by bringing critical advanced technologies into the farmland practices.

3D Bioprinting

3D Bioprinting is a form of additive manufacturing that uses cells and other biocompatible materials known as bioinks, to print living structures layer-by-layer which mimic the behavior of natural living systems. Three dimensional bioprinting is the utilization of 3D printing–like techniques to combine cells, growth factors, and biomaterials to fabricate biomedical parts that maximally imitate natural tissue characteristics.

Bioprinting (also known as **3D bioprinting**) is combination of **3D printing** with biomaterials to replicate parts that imitate natural tissues, bones, and blood vessels in the body. It is mainly used in connection with drug research and most recently as cell scaffolds to help repair damaged ligaments and joints. In this paper, we are looking at natural tissues related to food crops like rice, wheat or maize.

METHODOLOGY and the ARCHITECTURE

A Quantum Generator device has one or more Compute Units. A work-group executes on a single Compute unit. A Compute Unit is composed of one Processing Element and Seed Object. A Compute Unit may also include filter Units that can be accessed by its processing elements.

A Device is a collection of Compute Units. Quantum Generator device typically corresponds to a collection of multiple Compute Units generated by the seed of a number.

A Seed is a function declared in a program and executed on a quantum generating device. A seed is identified by the Seed Qualifier applied to any function defined in any program.

A Seed Object encapsulates a specific seed function declared in a program and the argument values to be used when executing this Seed Function.

A Synchronization refers to mechanisms that define the order of execution and the visibility of operations between two or more units of execution. The Operations are that define order controls in a program. They play a special role in controlling how operations of in one unit of execution (such as work-items) are made visible to another. Synchronization essentially involves establishing a relation between operations in two different units of execution that define an order control in a device.

Seed Objects

A seed is a function declared in a program. A seed is identified by the seed qualifier applied to any function in a program. A Seed Object encapsulates the specific seed function declared in a program and the argument values to be used when executing this seed function.

Seed Objects are created for any seed functions in program that have the same function definition across all Compute Units for which a program has been built successfully in a device.

Converting Data Points to Structure

The first and most important step is to create the data generated by the Seed Object in the correct structure or format and data analytics are a great tool for summarizing and analyzing large amounts of data. Before attempting to create data analytics, it is necessary to lay out data in the right structure as this can be of great help in creating detailed analytics as well reports that present the data in a clear and simple format.

The first step is to creating a table format in setting up the data in the correct table structure or format. This is the source data we will use when creating a table and the entire source data should be setup in a table layout.

The following is a list of components of a data table.

- **Fields** – Columns that define the values in the rows.
- **Column Header –** Name that describes the data in the field.
- **Data Records** – Rows in the table below the header that contain the data.
- **Record Set** – One row of data that contains values for each field.

The data table contains a column for each field and rows for each data record. The column fields are named with descriptive attributes that define the values in the record sets (rows). These are the descriptive fields that define what values will be in each row of the table. It's also important to note that field names (column headers) must be unique throughout the table.

In the structure above, we could start filtering and basically filters the table based on criteria we specify in the filter fields. With this format we could easily sum the column to produce the Total or create a product of columns with a different criterion or combination of them.

The basic rule of the data structure is that all values of the same type need to be in one column. This one rule should make it easier to quickly determine if the data is in the right structure.

As a part of the data generated by the Seed Object for the values for each field are derived from different parameter values along with sample cell images obtained from the Internet. All the defining characteristics of the cell images which are representative of structure of crop must be entered in each record set (row).

Concatenating or Merging two Structures

We would like to merge two structures into a new structure containing all the features of the two original structures. The main aim here is to create structures that are identical to the natural structure that are found in the tissues in the crops. We look at means for creating cell types which will be then self-assembled to create building blocks of natural tissues with independent arrangement and patterning to provide the required biological functions and architecture.

There is no direct ability in MATLAB that can be used to concatenate structures. However, there are different methods that can be used to merge structures in MATLAB.

A Color Image Merging Algorithm Using MATLAB

We present an image processing algorithm to merge two color images using MATLAB. The algorithm is based on the fact that only one color plane is used to detect similarities between images before the merging is accomplished.

Image merging of two images into a single image can be done using different methods. We will describe an efficient and simple procedure to produce a large merged image using MATLAB. The overall goal is to acquire several small and more detailed images of large objects and then compose a larger image file by combining these small image files. It is understood that large objects cannot be imaged with any great detail. The algorithm references two different images to be merged. The first image is a database image. The second image in the algorithm is the next image to be added to the database image. The algorithm starts by extracting a single color plane from both color images. The next step is to sample the pixel intensity values in the second image and to store in a matrix or table form the pixel intensities within the reference point. The contents of this matrix are then used to find a similar arrangement of pixel intensities in the database image. The algorithm starts at the left uppermost pixel in the database image. Then it takes the absolute value of the pixel differences between a single color plane, in a database image matrix, and the value matrix acquired by sampling a piece of the second image. These differences are stored into another matrix for future processing. After these values are calculated, the window moves to the very next pixel in the image. This process continues until all pixels in the image are analyzed. After every pixel position has an effective difference associated with it, the minimum difference is located in the difference matrix. Once this pixel position is known, the two images are merged using this specific point as a reference. Results show that color images can be successfully and efficiently merged.

The QG System

Our objective is to build a target system, we need to generate the cell for the device by running synthesis and implementation on the design. The cell includes custom logic for every Compute unit in the cell container. The generation of custom compute units uses the High-Level synthesis tool, which is the computer unit generator in the application compilation flow. Therefore, it is normal for this step to run for longer period of time than the other steps in the system build flow.

After all compute units have been generated, these units are connected to the infrastructure elements provided by the target device in the solution. The infrastructure elements in a device are all of the memory, control and output data planes which the device is formulated to support an application. The environment combines the custom compute units and the base device infrastructure to generate a cell binary which is used to program the QG device during application execution.

The processing flow of application execution is given as below:-

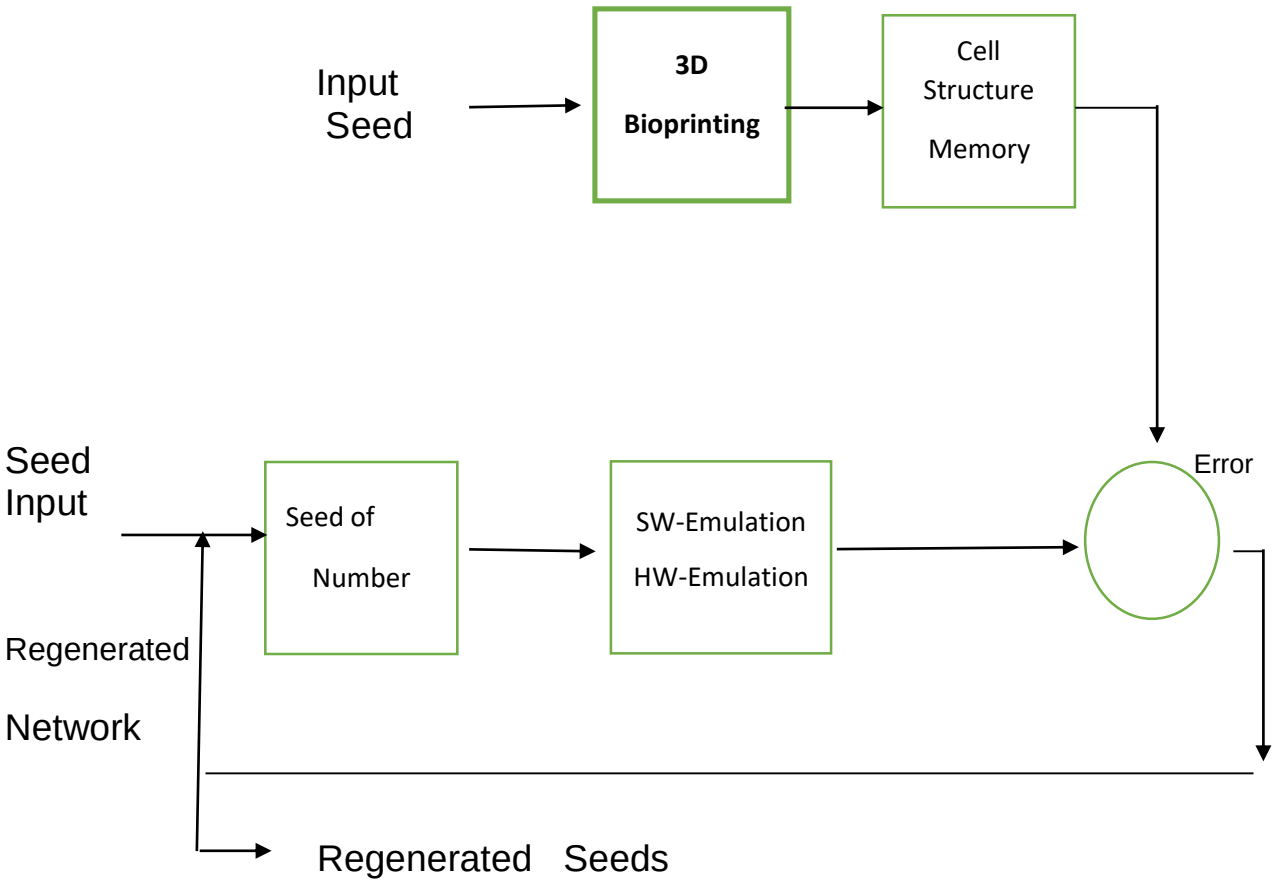

Fig. 1 Process Flow in a Quantum Generator.

The different steps in application are as below:-

1. 3D print a seed and copy its cell structure to memory.
2. Input seed with a seed of a number required.
3. Generate a seed kernel once.
4. Compare the kernel with 3d printed cell
5. If error in seed structure, generate the kernel again.
6. Repeat many times till the seed number is met.

TEST RESULTS

This paper presents an efficient and simple procedure to produce a large merged image using MATLAB and the goal is to acquire several small and more detailed images of large objects and then compose a larger image file by combining these small image files. Results show that color images can be successfully and efficiently merged into a larger image.

CONCLUSION

Quantum Generators (QG) creates new seeds iteratively using the single input seed and the process leads to a phenomenon of generating multiple copies of kernels in repetition. We presented a method to produce natural tissue structures of crops that are constructed with appropriate biological properties. Results show that the proposed method performs well with sample images obtained from the Internet and the results suggest that it is possible to achieve self-assembled biological blocks of pattern with required architecture.

REFERENCE

1. E Boyer: “A Color Image Merging Algorithm Using Matlab” https://peer.asee.org › a-color-image-merging-algorith.
2. Poondru Prithvinath Reddy: “Quantum Generators: Pattern Analysis for 3D Model Reconstruction from the Structured Compute Units”, Google Scholar.