BUBBLESRNG

A TRUE RANDOM NUMBER GENERATOR USING AIR BUBBLES AS AN UNCONVENTIONAL SOURCE OF ENTROPY

USER MANUAL

Contents

SOFTWARE REQUIREMENTS

Operating System: Microsoft Windows 7, 8 or 10, 64-bit versions. May run on other versions of Windows but at the time of writing these are the only platforms on which BubblesRNG has been tested. The Java Runtime Environment (JRE v6 or better) must also be installed.

File Dependencies: The file opencv_java2411.dll needs to be copied to a folder included in the *path* environment variable such as *C:\windows\system32* or to the folder from which BubblesRNG.jar is executed. The DLL can be extracted from bubblesrng.jar using 7-zip, WinRAR or similar. Alternatively, it can also be downloaded as part of a package from the OpenCV website;
(http://sourceforge.net/projects/opencvlibrary/files/opencv-win/2.4.11/opencv-2.4.11.exe/download)

We have also included an MSI installer which can be used to install BubblesRNG. The installer creates a *C:\bubbles* folder to which it extracts the bubblesrng.jar, opencv_java2411.dll and the getrnd.jar utility. A shortcut is also created on the desktop which is used to launch BubblesRNG. If installed, there is no need to copy opencv_java2411.dll since the installer takes care of copying it c:\bubbles from where bubblesrng.jar is executed.

HARDWARE REQUIREMENTS

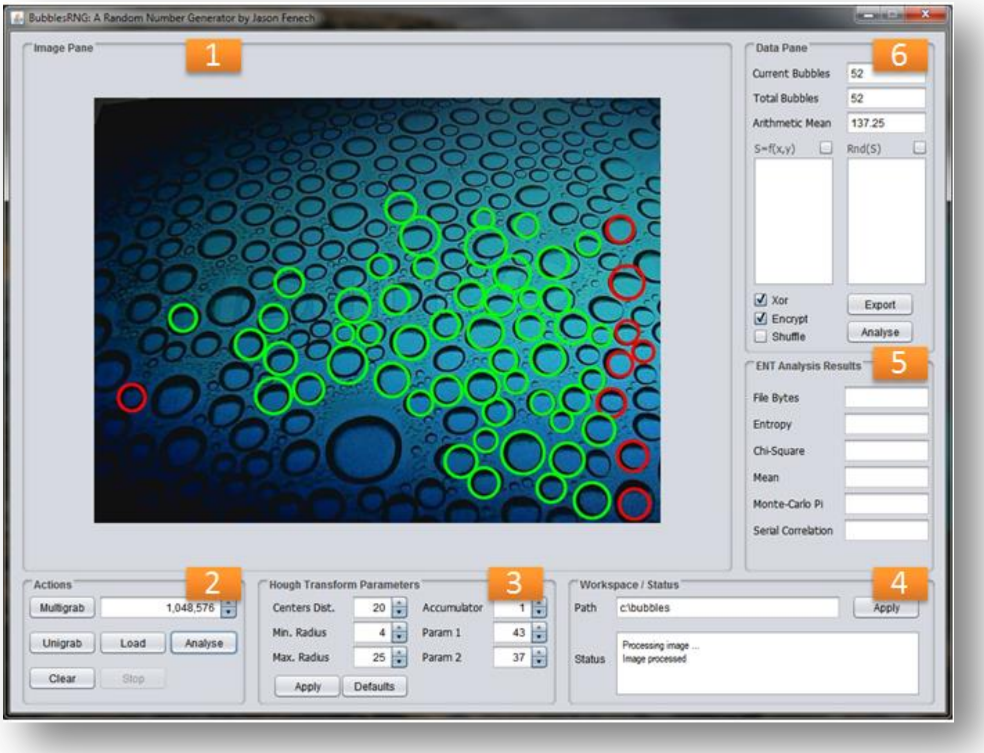
Webcam: A minimum resolution of 640x480 pixels @ 30FPS. Manual focusing is also important to ensure sharp images are captured unless this is satisfactorily handled through automatic focusing.

RAM: A minimum of 4GB providing sufficient memory for the javaw.exe process.

Entropy source: Ideally the same entropy source outlined in this project should be used to replicate the non-deterministic properties observed. However, during the course of this project, we identified other sources which could be used even though they could prove to be less secure. One such example consists in pointing the webcam towards a monitor playing random videos (https://goo.gl/inSF1e). For this to work, the CHT parameters are tweaked until the desired results are achieved. The “Encrypt” option should also be enabled.

THE GRAPHICAL USER INTERFACE

The interface is sub-divided into 6 sections or panes, the function of each explained as follows;



BubblesRNG: A Random Number Generator by Jason Fenech

Current Bubbles: 52
Total Bubbles: 52
Arithmetic Mean: 137.25
☑ Xor
☑ Encrypt
□ Shuffle

Multigrab: 1,048,576

Centers Dist.: 20
Min. Radius: 4
Max. Radius: 25
Accumulator: 1
Param 1: 43
Param 2: 37

Path: c:\bubbles
Status:
Processing image ...
Image processed

1. *Image Pane*

   This is where captured and processed images are displayed. It allows users to visually follow the capture and bubble detection process.

2. *Actions Pane*

   This pane is populated with a series of buttons and a spinner text fields. The buttons and spinner text fields map to the following actions;

   - **Multigrab**: Initiates the "Multiple Grab" process (Section 2.2.3). The generator keeps on capturing images until the number of bytes generated match the value entered in the "number of bytes to be generated" text field.

   - **Unigrab**: Initiates the "Single Grab" process as explained in Section 2.2.3.

   - **Load**: This allows a user to load a JPG image from disk. This can be useful when a user needs to experiment no hardware supplementing the required entropy is available.

   - **Analyse**: Calls the bubble detection process on the image just captured. This button needs pressing only during the "Single Grab" process as the method it calls is called automatically during the "Multiple Grab" process.

    - **Clear**: Clears all the value displayed in panes 4, 5 and 6 as well as re-setting some data structures to their initial state as implemented in code. The image pane is cleared too.

    - **Stop**: This is used to exit the "Multiple Grab" loop.

  - The spinner text field value is set to 1MB (1048576 bytes) by default. The user can increase or decrease this value in steps of 256 bytes using the spinner controls or by simply typing in the desired value. The value corresponds to the number of bytes to be generated during the "Multiple Grab" process.

3. *Hough Transform Parameters Pane*

   This pane is populated with six spinner text fields each corresponding to one of the user-definable CHT parameters outlined in Section 2.1.3. They are set to the default values used during the project's testing phase but can be changed as required. The "Apply" button turns green to signal that one or more parameters have changed. Pressing it, commits the changed values to the generator. Parameters can also be changed while the generator is actually running. The "Defaults" button resets the values to their default.

4. *Workspace / Status Pane*

   This pane serves two functions. It allows a user to change the default workspace, i.e. the folder under which BubblesRNG writes the captured images, number sequences and test results. The default path is set to "c:\bubbles". If this is changed, the "Apply" button turns green to remind the user to press the button to commit the change. Pressing the button creates a new folder, unless it already exists, to which it copies the ENT utility.

   The "Status" textbox is used as a notification area giving the user feedback about any ongoing process, errors and such.

5. *ENT Analysis Results Pane*

   The results from the ENT analysis are displayed here. The text fields are populated after running ENT either manually or automatically.

6. *Data Pane*

The components on this pane serve multiple purposes.

1. The "Current Bubbles" text field returns the number of bubbles detected in the current image.

2. The "Total Bubbles" text field keeps a count of all the bubbles detected during a "Multiple Grab" process.

3. The "Arithmetic Mean" returns the mean value for the numbers generated by analysing the current image.

4. Enables or disables the displaying of data in the scroll pane labelled 6. (GUI may perform sluggishly if enabled)

5. Enables or disables the displaying of data in the scrollpane labelled 7.

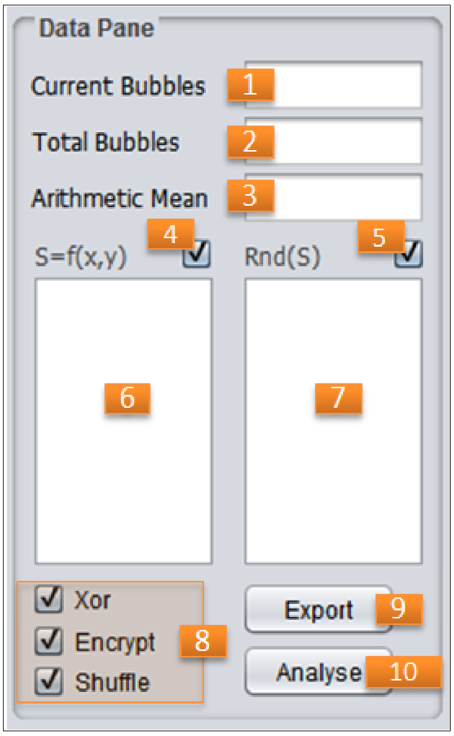
6. Displays a list of the raw data generated.

7. Displays a list of conditioned data.

8. This group of checkboxes controls the output of the RNG. (See Section 2.3.3 for full details).

9. When the "Export" button is pressed, the generated sequence of random numbers is written to disk under the workspace folder.

10. The "Analyse" button turns green after a sequence has been exported. Pressing it runs the ENT tool against the sequence thereby populating the text fields in the Ent Analysis Results pane with tests results. If ent.exe is missing, an error is displayed in the Status window.



RUNNING BUBBLESRNG

BubblesRNG can be executed by double-clicking on the JAR file using Explorer in Microsoft Windows or by running it in a command prompt (cmd.exe) using the command ***java –jar BubblesRNG.jar***.

Note*:* The software will fail to start if the OpenCV DLL dependency is missing as shown in this output;

```
C:\>java -jar bubblesRNG.jar
Exception in thread "AWT-EventQueue-0" java.lang.UnsatisfiedLinkError: no opencv_java2411 in
java.library.path
        at java.lang.ClassLoader.loadLibrary(Unknown Source)
        at java.lang.Runtime.loadLibrary0(Unknown Source)
        at java.lang.System.loadLibrary(Unknown Source)
        at bubblesRNG.GUI$1.run(GUI.java:103)
        at java.awt.event.InvocationEvent.dispatch(Unknown Source)
        at java.awt.EventQueue.dispatchEventImpl(Unknown Source)
        at java.awt.EventQueue.access$500(Unknown Source)
        at java.awt.EventQueue$3.run(Unknown Source)
        at java.awt.EventQueue$3.run(Unknown Source)
        at java.security.AccessController.doPrivileged(Native Method)
        at java.security.ProtectionDomain$1.doIntersectionPrivilege(Unknown Source)
        at java.awt.EventQueue.dispatchEvent(Unknown Source)
        at java.awt.EventDispatchThread.pumpOneEventForFilters(Unknown Source)
        at java.awt.EventDispatchThread.pumpEventsForFilter(Unknown Source)
        at java.awt.EventDispatchThread.pumpEventsForHierarchy(Unknown Source)
        at java.awt.EventDispatchThread.pumpEvents(Unknown Source)
        at java.awt.EventDispatchThread.pumpEvents(Unknown Source)
        at java.awt.EventDispatchThread.run(Unknown Source)
```

A webcam must be connected to the computer on which BubblesRNG will be run. Failing this, none of the available features will work. A “No camera detected” error message is displayed whenever a webcam is not detected or has been unplugged.

PERFORMING A SINGLE GRAB OPERATION

1. Make sure a web camera is plugged in.

2. Click on the "Unigrab" button. The captured image should be displayed in the **Image Pane**. Check the status window for the *event "Grabbed c:/bubbles/captImage.jpg".*

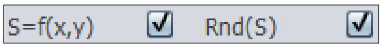
3. Click on the "Analyse" button to detect any bubbles in the present image. Check the status window for the events *"Processing image ... Image processed".*

4. Every bubble detected is either circled in green (valid) or red (discarded) on the processed image which is displayed in the **Image Pane**.

   Note: Steps 1-4 verify that the image capturing and detection processes are functioning correctly.

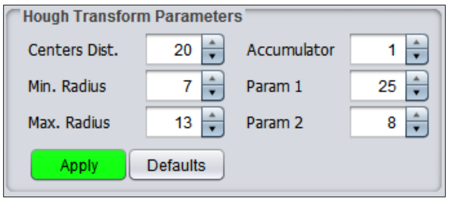
5. Press the "Clear" button to reset the interface.

6. Tick on check-boxes 4 and 5 on the **Data Pane**.



S=f(x,y) ☑ Rnd(S) ☑

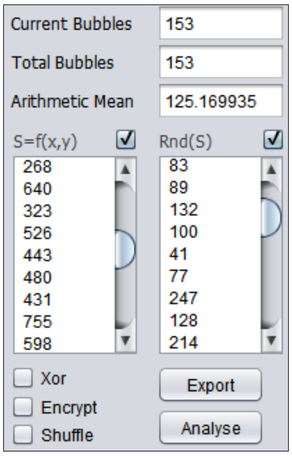
7. Change any of the Hough Transform Parameters and press the "Apply" button.



Hough Transform Parameters

Centers Dist. 20
Min. Radius 7
Max. Radius 13
Accumulator 1
Param 1 25
Param 2 8

8. Press the "Analyse" button from the **Actions Pane**. Raw and conditioned data should now be visible in each of the respective scroll panes on the **Data Pane**. You can scroll up or down to view the entire lists.



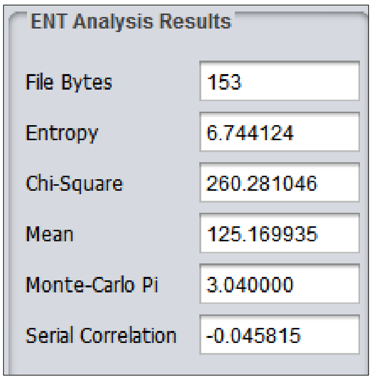
Current Bubbles 153
Total Bubbles 153
Arithmetic Mean 125.169935

S=f(x,y) ☑	Rnd(S) ☑
268	83
640	89
323	132
526	100
443	41
480	77
431	247
755	128
598	214

□ Xor
□ Encrypt
□ Shuffle

9. Press the "Export" button on the **Data Pane.** In the status window you should see an event similar to "*File exported to c:/bubbles/bubblesHRNG_270815_054513_---.bin".* The file represents the sequence of random numbers written to disk in binary format. The 3 characters preceding the filename extension (.bin) correspond to the RNG options (X, E, S) used to generate the sequence.

10. The “Analyse” button should turn green after the export process completes. Press the “Analyse” button to run a statistical test on the sequence generated using ENT.



ENT Analysis Results

File Bytes	153
Entropy	6.744124
Chi-Square	260.281046
Mean	125.169935
Monte-Carlo Pi	3.040000
Serial Correlation	-0.045815

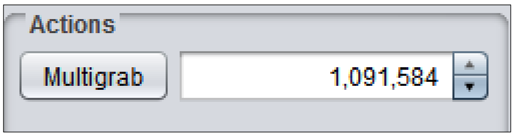
PERFORMING A MULTIPLE GRAB OPERATION

1. Make sure a web camera is plugged in.

2. If selected, untick checkboxes 4 and 5 on the **Data Pane.** This reduces processing time by minimizing the number of times the UI is refreshed due to scrolling.

3. Select the RNG options by ticking the corresponding checkboxes on the **Data Pane**.

4. The length of the generated sequence (bytes) can be changed by inputting a new value in the corresponding text field on the **Actions Pane**. The value should be **2^n**.



1,091,584

5. Adjust the CHT parameters if required. Setting “Param 2” to a low value will increase the number of false positives detected (see Section 2.1.3).

6. Press the “Multigrab” button to start the capture process. Most of the remaining controls on the interface are ghosted out during this process which terminates once the number of bytes generated matches that specified by the user. The process can also be stopped any time by pressing the “Stop” button on the **Actions Pane**.

   Notes:

   a. The CHT parameters can be changed on the fly during the process requiring the user to press “Apply” to commit the changes. This may be used whenever the arithmetic mean is seen to deviate significantly from the ideal value of 127.5.

   b. The RNG options can be changed on the fly as well. Again the Arithmetic Mean value is an indicator of the how well the number generated are distributed on the interval [0,255].

   c. When the process ends, both the “Export” and “Analyse” functions are executed automatically.

FILES GENERATED BY BUBBLESRNG

Under the workspace path folder you will find any of the following files;

1. *Ent.exe* – This is the ENT executable, copied to the default or user-specified workspace whenever BubblesRNG is first run or after the "Apply" button on the **Workspace / Status Pane** is pressed.

2. *bubblesHRNG_<date>_<time>_<RNG Options>.bin* – This is a binary file corresponding to the sequence of random numbers generated;

   - <date> - ddmmyy
   - <time> - hhmmss
   - <RNG Options> - {**X**or-**E**ncrypt-**S**huffle}

   Ex. *bubblesHRNG_270815_014831_XE-.bin* – a sequence produced on the 27th Aug 2015 @ 01:48:31 using the Encrypt and XOR options.

3. *ENT_CSV_<date>_<time>.txt* – is a text file containing the statistical results in comma delimited format generated by the ENT for the sequence with matching <date> and <time> values in the filename.

   Ex. *ENT_CSV_270815_014831.txt* is the results file for sequence contained in the file *bubblesHRNG_270815_014831_XE-.bin.*

4. *ENT_Full_<date>_<time>.txt* – a text file containing the statistical results in full format generated by the ENT for the sequence with matching <date> and <time> values in the filename.

5. *captImage.jpg* – The image last captured by the web camera.

6. *procImage.jpg* – This is a clone of captimage.jpg but in which detected bubbles are circled in green or red.

MEMORY ISSUES

BubblesRNG makes use of a few 3rd party libraries one of which written in C++ and handled via a Java wrapper. There are instances where the Java garbage collector is unable to free memory resources allocated to C++ objects or data structures. Ideally one should monitor the javaw.exe process through task manager to see how the system is performing. The software triggers the GC every 50 images taken. Nevertheless, BubblesRNG may cause OS performance issues on system with 4GB of RAM or less.