An Easy To Build Reflectance Transformation Imaging (RTI) System

Ted Kinsman

The article discusses the construction a simple reflectance transformation imaging (RTI) setup using an Adurino microprocessor and a 3D printer.

Introduction

It can be sometimes difficult for a single photograph to adequately record and represent all of an object's important detail. By using reflectance transform imaging (RTI) photographic techniques, researchers are better able to enhance an object's individual topography, texture, and color. Unlike one single photograph, an RTI photographic system may utilize numerous lighting angles to produce sequential images of the same object or artifact that may reveal obscured or hidden detail.

The RTI technique provides a valuable tool for scientists and conservators for the research and preservation of important historical fine art collections. RTI enhanced images prove invaluable for both conservation and fine art authentication purposes. RTI systems are also used in the area of forensic science to photographically record and analyze bullet or projectile striations. Additionally, RTI may be an extremely useful tool for imaging and analyzing pottery fragments, small archaeological tools, and stone artifacts. RTI applications are limitless, and they may provide key visual information for researchers in many investigative areas.

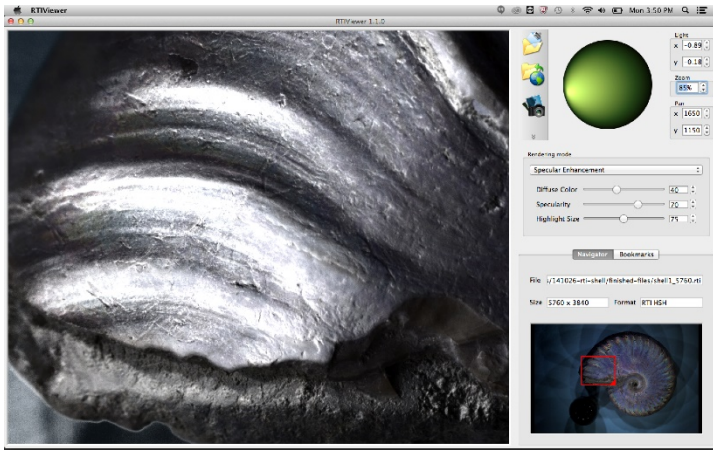

Figure 1. *The RTIViewer program shows the close-up of an ammonite fossil. The specular highlights can be adjusted. The lighting angle is controlled by the position of the curser in the green circle. Desired images can be exported in full resolution.*

Botanical and zoological fossils also represent ideal subjects for RTI imaging as many of these ancient fossils have intricate shapes, repeating patterns, and interesting surface textures. Ammonite fossils in particular, have ribbed, spiral-form shells (Figures 1–4). Ammonites were free-swimming molluscs found in our oceans, living about the same time as the dinosaurs.

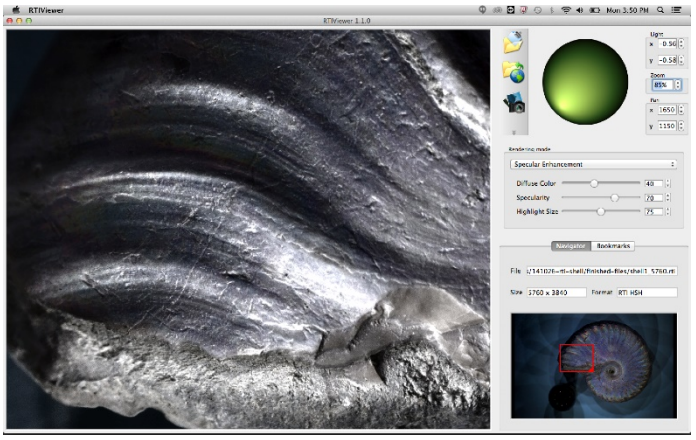

Figure 2. *The RTIViewer program shows the close-up of an ammonite fossil. The specular highlights can be adjusted. The lighting angle is controlled by the position of the curser in the green circle. Desired images can be exported in full resolution. The location of the light source is different than Figure 1.*

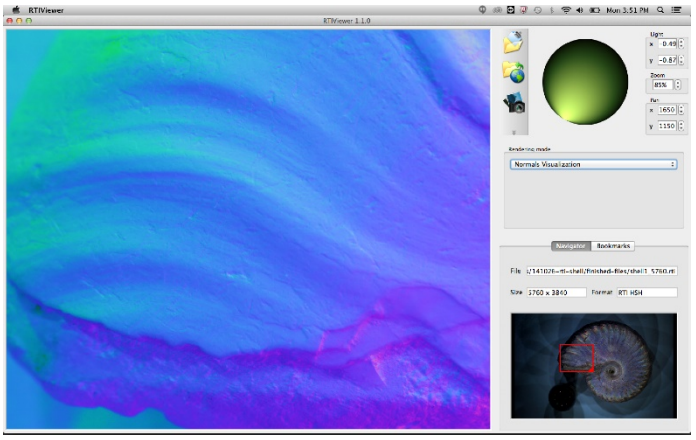

Figure 3. *The RTIViewer program shows the close-up of an ammonite fossil. The surface is imaged in the normal vector*

mode. The different colors represent the direction a particular point of the surface is pointing. This viewing mode can often highlight slight imperfections in the surface of the subject.

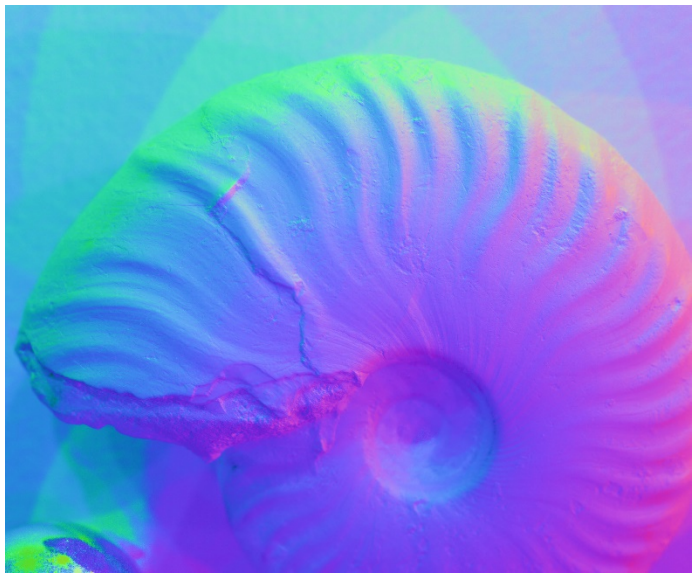

Figure 4. *The RTIViewer program shows an ammonite fossil imaged in the surface normal vector mode. The different colors represent the direction a particular point of the surface is pointing.*

Methods

Constructing the RTI System

RTI is a computational photographic method that collects between 40 and 100 images with light sources at different fixed locations. Once the software is able to deduce the location of the light sources, the resulting images can be used to create a mathematical 3D map of the surface. Software can calculate surface normal vectors from the images. Subsequently, surface features can then be exaggerated to show the exquisite detail. The RTI technique is dependent on taking a number of images with light sources at fixed locations.

The process described in this paper outlines a simple device that automatically collects reference images. The software has been developed by a number of researchers and is hosted by Cultural Heritage Imaging's website (http://culturalheritageimaging.org/Technologies/RTI/). Photographers, researchers, and other interested parties are encouraged to review the technique. The construction and viewing of the RTI images is done with the RTI builder software developed by Cultural Heritage Imaging, and the resulting RTI file is viewed with their RTI viewer software.

The requirement of the light sources for the RTI image set is that they be approximately 50 in number, and placed evenly on a half sphere. One readily available dome was a simple lighting dome. In order to remove unwanted room light, the inside was spray painted black and then 4 rows of holes were drilled, evenly spaced from the horizontal. Each row holds 12 individual lights, which brings the total lights to 50 in number. Each hole was numbered to make the wiring easier. The large circular hole at the top of the dome is for the camera and was cut with a Dremel Drill. This technique easily removes the unwanted plastic and makes a smooth cut. The lights are large 10 mm white LEDs, which draw 80mA. These can easily be powered by the 5V DC, supplied by the microprocessor. To control each LED, an LED light array, manufactured by Adafruit Industries, was used. This array has the advantage of limiting the number of control wires needed to drive the 50 individual LEDs.

The wiring schematic is simple, but the execution is quite time consuming. It took about 12 hours to cut and solder all the contacts – this was slower than usual due to the Teflon coated wire, which was difficult to strip. Future systems will be assembled with a wire wrap tool, which is preferred for prototyping circuits.

The idea for the program is relatively simple too. Initially, an individual light is turned on, the camera shutter is triggered, and then the light turns off. The program sequentially repeats the cycle for all the remaining lights. The duration of the LED light is adjusted within the software. The prototype is designed for a 105mm macro lens and a Canon 5D Mark III DSLR camera body. In order to image at a low ISO and an adequate depth of field, the lights are programed to stay on for 5 seconds. This allows the full set of images to be collected in approximately 4 minutes. The exposure can be optimized for the subject and increased if desired. If a large number of RTI image sets are being collected, the LED 'on' time should be reduced to just slightly longer than the exposure time. To insure repeatability and exposure efficiency, it is important that the LED illumination continues for the complete duration of the shutter exposure.

The Schematic

As the Adurino microprocessor does not have 50 outputs to control the lights, the circuit has to use an LED lighting array. To control the array, the microprocessor communicates with an LED array driver (board), which in this case is the Adafruit 16x8 LED Matrix Driver Backpack (HT16K33 Breakout). This board is currently available from Adafruit for about $8.00 USD. The communication protocol for this board is I2C, which uses only two pins. As a result, there may be up to eight selectable I2C addresses for a total of eight matrices, each one controlling 16x8 LEDs for 1024 total LEDs. This matrix driver and the microprocessor are both available at Adafruit Industries at http://www.adafruit.com/

This board is designed to work specifically with the Arduino Uno microprocessor and an array of LED lights. After stringing long lines from each LED back to the array control, a shorter technique was discovered in which the LEDS in each row can be connected together on their ground side before returning to the array driver. This simplifies the wiring, but complicates how to visualize the circuit. This wiring set-up can be seen as the blue wires on the prototype dome (Figures 5, 6, 7).

The camera is triggered by controlling pin 12 on the

microprocessor. When this pin goes to 5 Vdc, the reed switch will trip the camera. The reed switch is used to isolate the camera trigger from any voltages generated by the microprocessor. This is a safety measure to ensure the camera is not exposed to any dangerous voltages. The reed switch is also easy to wire and is very inexpensive. A schematic of a trigger system for the camera has not been included here, since most are different.

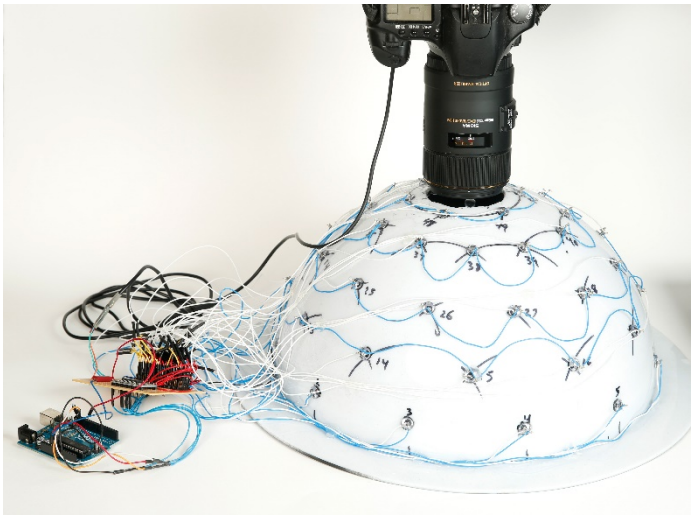

Figure 5. The placement of the camera on the first prototype Reflectance Transformation Imaging (RTI) sphere. *The Adruino Uno microprocessor drives an array of LEDs. Each LED is independently controlled to take a set of 50 images that are in turn combined to an RTI file.*

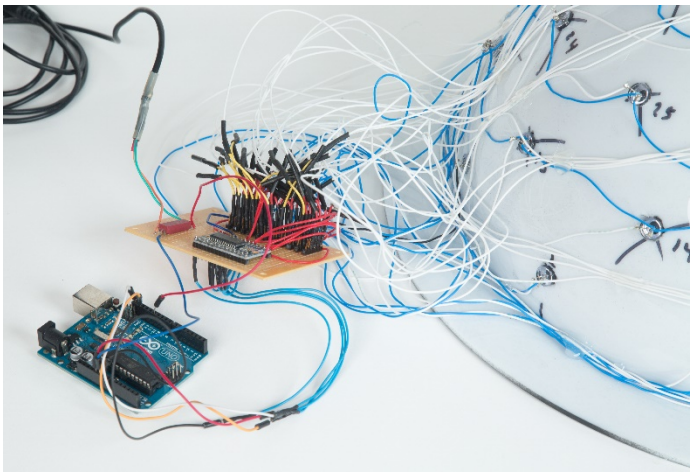

Figure 6. Close-up image of the wiring on the prototype RTI system.

Creation of RTI File

The resulting image set is converted to JPEG image format (.jpg) and run through the RTI builder software. The software is free to the public and the easy-to-follow directions are available in PDF format on the Cultural Heritage Imaging website. To view the resulting file, the RTI viewer program is used. This program is also available free of charge on the Cultural Heritage Imaging website.

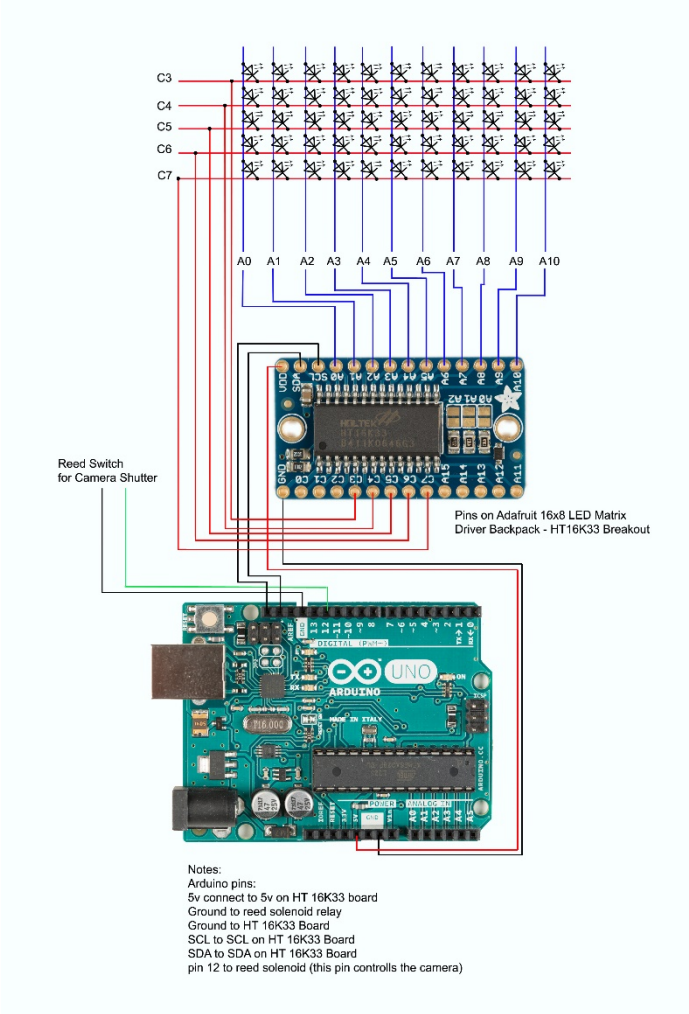

Figure 7. The wiring schematic for setting up the LED array and connecting the Arduino Uno microprocessor.

3D Dome

The creation of the dome presented a number of problems. To simplify the process and make the design available to researchers, a 3D dome was designed that can be printed in a tabletop 3D printer. The bed size of most 3D printers is limited, so this dome is designed for macro work. The holes are sized to be about the size of a 5mm LED. The holes will change diameter as the RTI dome is scaled for a particular printer. The holes can be considered as pilot holes and can be easily drilled to the desired diameter. This dome also includes an excess of holes, so that future systems can use sets of LEDs, each operating at a different wavelength (Figures 7, 8).

I hope that you find this RTI system an efficient and elegant technique for collecting images for the RTI process. I would like to thank Andy Kinsman for his help with the

design of the 3D dome file.

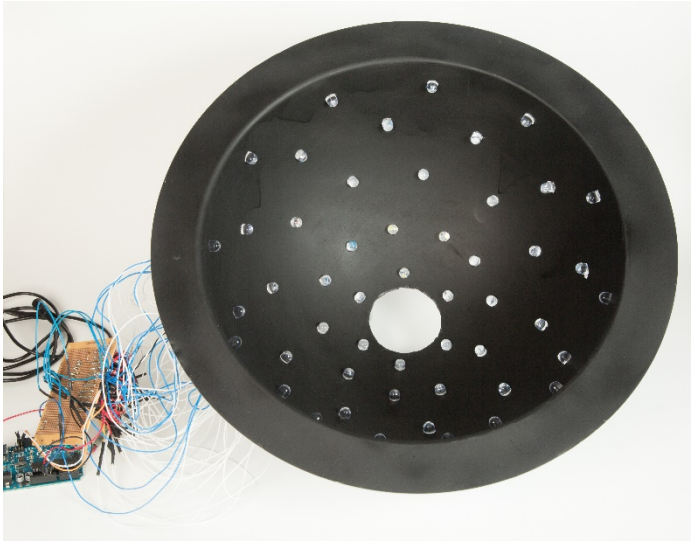

Figure 8. Inside of the large sphere showing the 50 LED lights. *Each light is a 10 mm supper bright LED and can be independently controlled by the Adruino Uno microprocessor.*

Figure 9. The 3D printed dome for 5mm LED lights. *The computer file is available for download and experimentation.*

Programming Code for RTI LED Dome:

```
// long lead on LED goes in A0-A15 holes.

#include <Wire.h>
#include "Adafruit_LEDBackpack.h"
#include "Adafruit_GFX.h"

Adafruit_8x16matrix matrix = Adafruit_8x16matrix();

#define BUSY_LED   13     // Board IO pin of Arduino
ORANGE LED
#define CAMERA_CTL  12     // Board pin controlling the
Camera Trigger
                    // active low.

#define LED_COUNT   50     // number of LEDs to sequence
through

void setup() {
  Serial.begin(9600);
  Serial.println("RTI multi-spectral camera");

  matrix.begin(0x70);  // pass in the I2C address
  matrix.clear();     // clear all lights
  matrix.writeDisplay();

  // set the digital pin as output:
  pinMode(CAMERA_CTL, OUTPUT);
  digitalWrite(CAMERA_CTL, HIGH);  // active low

  // set up Arduino LED as output
  pinMode(BUSY_LED, OUTPUT);
  digitalWrite(CAMERA_CTL, HIGH);  // active low
}

/* array indexes - insert LED with long lead on A line.

15  A15 * * * * * * * * *
14  A14 * * * * * * * * *
    .  * * * * * * * * *
    .  * * * * * * * * *
    .  * * * * * * * * *
1   A1 * * * * * * * * *
0   A0 * * * * * * * * *
Y=     C7 C6 C5 C4 C3 C2 C1 C0
    X= 0  1  2  3  4  5  6  7
*/

void loop() {
  int x,y,count=0;

  for(x=0;x<8;x++)        // x's are the C lines
     for(y=0;y<11;y++)    // y's are the A lines
     {
        // toggle BUSY LED
        digitalWrite(BUSY_LED, HIGH);  // ON

        // turn on one LED
        matrix.drawPixel(x, y, LED_ON);
        matrix.writeDisplay();  // write the changes we just
made to the display
        delay(6000);   // this delay determines the length the
led is on

        // trigger the camera
        digitalWrite(CAMERA_CTL, LOW);  // active low
        delay(50);
        digitalWrite(CAMERA_CTL, HIGH);  // active low
        delay(50);

        // turn off the only LED on
        matrix.drawPixel(x, y, LED_OFF);
        matrix.writeDisplay();  // write the changes we just
made to the display

        // delay while image stored on camera?
        delay(1000);

        // toggle BUSY LED
        digitalWrite(BUSY_LED, LOW);  // OFF
        delay(100);                 // so we can see the OFF state

        // test for end of light count
        count++;
        if( count >= LED_COUNT) goto DONE;
     }

DONE:
  Serial.println("Camera sequence complete");
  while(1);   // infinite loop- stop here

}
```

Acknowledgements

Ted Kinsman would like to thank Andy Kinsman for his help with the design of the 3D dome file.

Resources

The 3D dome is available for download at:
emkpph.cias.rit.edu/RTI/rti-half-dome-130mm.stl

The Arduino uno code is available for download at:
emkpph.cias.rit.edu/RTI/RTIsequence.ino

Cultural Heritage Imaging (RTi) software is available at:
http://culturalheritageimaging.org/Technologies/RTI/

References

Cosentino, A., Stout, S., and Scandurra, C. 2015. Innovative Imaging Techniques for Examination and Documentation of mural paintings and historical graffiti in the catacombs of San Giovanni, Syracuse. *International Journal of Conservation Science*, 6(1): 23-34.

Cosentino, A. 2013. Macro Photography for Reflectance Transformation Imaging: A Practical Guide to the Highlights Method. *e-conservation Journal*, 1:70-85.

Pawlowicz, L. 2015. Creating a Portable Dome-RTI system for Imaging Lithics. Cultural Heritage Imaging, CHI Blog. May 26, 2015.

https://culturalheritageimaging.wordpress.com/2015/05/26/creating-a-portable-dome-rti-system-for-imaging-lithics/

Author

Ted Kinsman is an Assistant Professor of Scientific Photography at Rochester Institute of Technology. Mr. Kinsman can be contacted at: emkpph@rit.edu