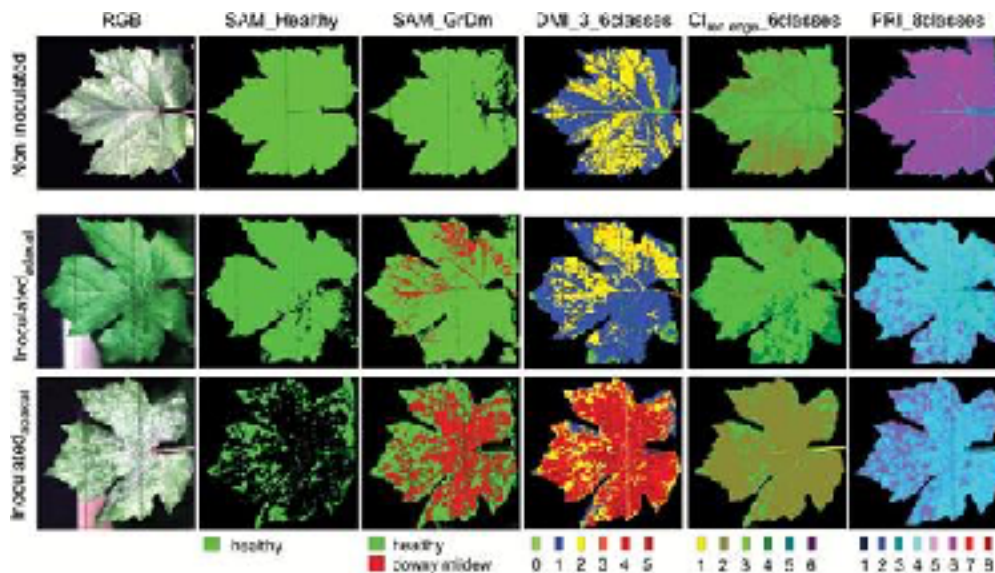


COLOR SPECTRUM RECONSTRUCTION USING NEURAL NETWORKS

THE GOAL

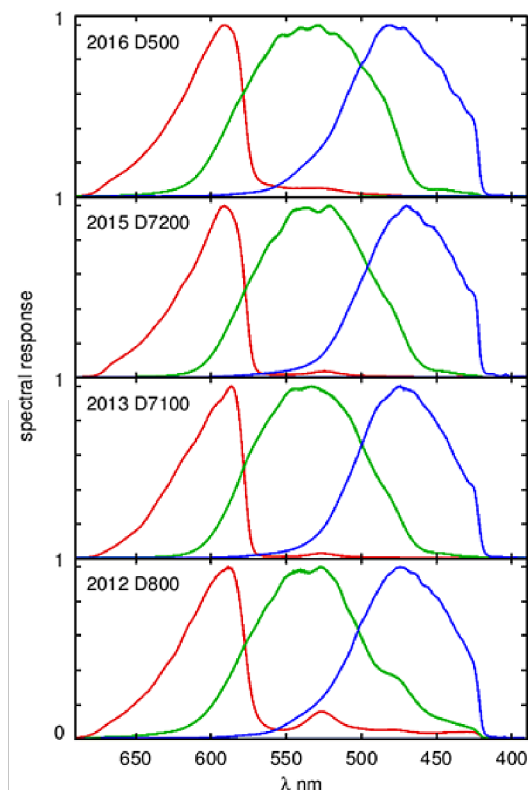
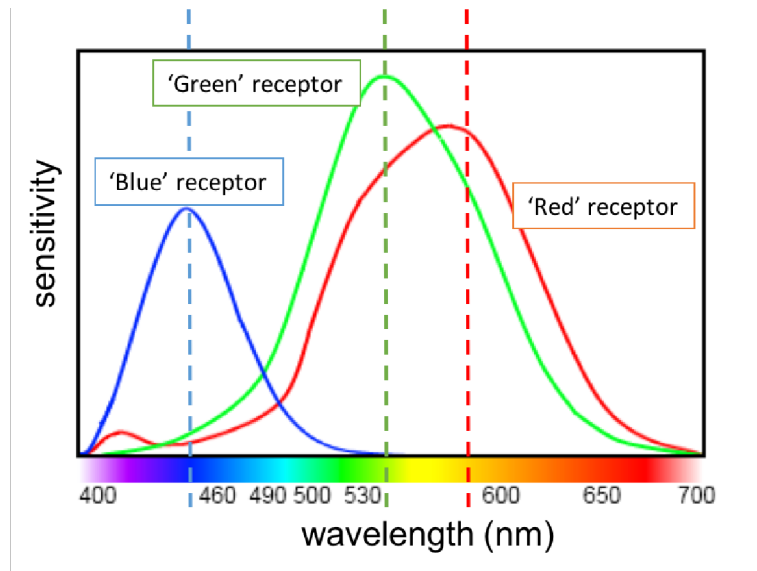
Spectral images tell us much more about natural objects than the usual RGB color images. By looking at the reflectance at specific wavelengths, we can see the reaction of plant leaves to infections, among other things.



Hyperspectral phenotyping of the reaction of grapevine genotypes to *Plasmopara viti*
 Erich-Christian Oerke Katja Herzog Reinhard Toepfer
 Journal of Experimental Botany, Volume 67, Issue 18, 1 October 2016, Pages 5529–5543, <https://doi.org/10.1093/jxb/erw388>

HYPERSPECTRAL IMAGING

SPECTRUM, COLOR VISION, RGB IMAGING



Spectral response of the human eye

<http://people.eecs.berkeley.edu/~cecilia77/graphics/a6/>

Spectral Ray Tracing Team members :

Cecilia Zhang, Ashwinlal Sreelal, Justin Comins

Spectral response for some Nikon c

Bernard Delley

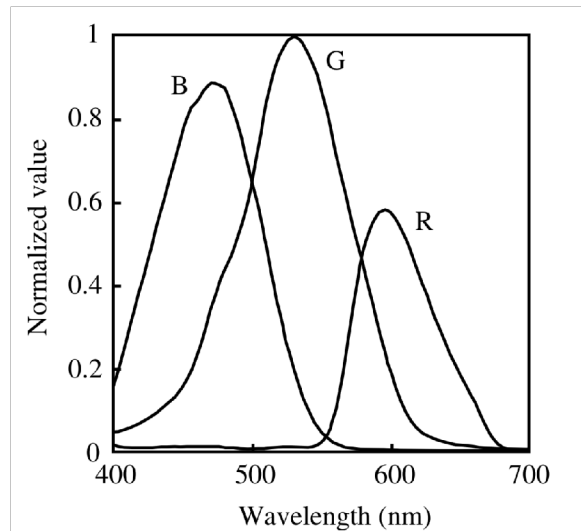
<https://www.dpreview.com/forums/th>

HYPERSPECTRAL IMAGING

SPECTRUM FROM RGB DATA

Color Signal Estimation of Surrounding Scenes from Human Corneal Surface Images

Ryo Ohtera, Shogo Nishi and Shoji Tominaga, Journal of Imaging Science and Technology 58(2): 020501-1-020501-12, 2014.



$$\rho_i = \int_{400}^{700} E(\lambda) R_i(\lambda) d\lambda + n_i \quad (i = R, G, B).$$

Wiener estimate (after σ is found):

$$\hat{\mathbf{e}} = \mathbf{C}_{ee} \mathbf{R}^t \left(\mathbf{R} \mathbf{C}_{ee} \mathbf{R}^t + \sigma^2 \mathbf{I} \right)^{-1} \boldsymbol{\rho}.$$

HOME-MADE HYPERSPECTRAL IMAGING

SPECTRUM FROM RGB DATA

We will try to re-create the color spectrum at each image pixel, from the RGB intensity components.

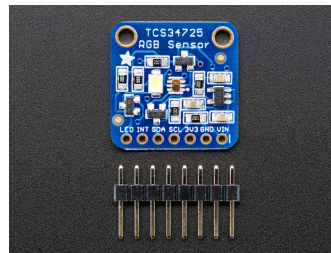
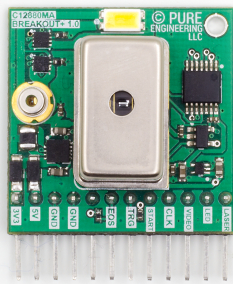
- As a first step, we collect data sets of spectral reflectivity S_i together with RGB sensor values C_i . Here each $S = \{s^1, s^2, \dots, s^n\}$, that is, n spectral bands; and $C = \{c^R, c^G, c^B\}$, the 3 color sensor values.
- We train a neural network to generate $\{S_i\}$ from $\{C_i\}$.
- Then, hopefully, we can scan a color image and replace each pixel with n spectral values, thus generating n hyperspectral images, on the **cheap!**

HOME-MADE HYPERSPECTRAL IMAGING

COLLECTING SPECTRAL AND RGB DATA

We use two sensors:

1. Hamamatsu C12880MA MEMS u-Spectrometer and Breakout Board
2. Adafruit RGB Color Sensor with IR filter and White LED - TCS34725



The Hamamatsu spectrometer board has a white LED (on top) and an ultraviolet laser (left side). We use this LED for illumination. The RGB sensor also has an LED, we might try comparing their spectrum. We have also tested using a krypton incandescent bulb (100 VAC, 40W), which has probably better spectral properties, but it is bulky, hot, and requires 100VAC, motivating the switch to the LED.

We have used 164 pieces of 120 * 120 mm size colored origami papers as samples. This is a rather poor choice, as the printing ink might not have sufficiently rich spectral variation; please somebody find a better sample set!

HOME-MADE HYPERSPECTRAL IMAGING

SENSOR PROGRAM

A simple Arduino program interfaces with the Hamamatsu sensor, driving its acquisition interface; and with the RGB sensor, using its I2C bus interface. From Arduino we send the sensor data through the “Serial” USB interface to a PC, each sample a line of text. By looking at the sensor data from a serial terminal program, e.g. “screen”, we can verify the correct operation. The same interface also accepts commands to turn the LED or the laser ON/OFF.

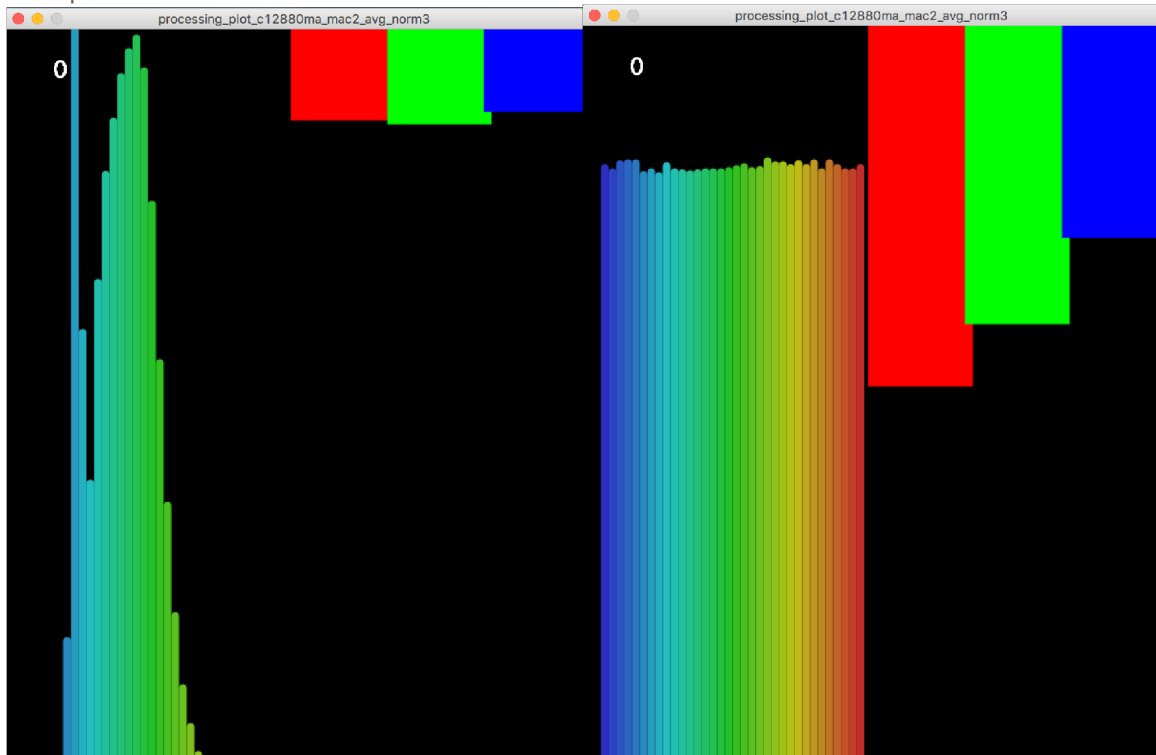
To actually perform the data collection, we use a Processing program, which displays graphically the current spectrum and RGB values, and save the data in a text file.

HOME-MADE HYPERSPECTRAL IMAGING

PROGRAM IN THE “PROCESSING” LANGUAGE

To reduce the noise and to make it easier for the neural network, we average 8 spectral bands into one, giving a final 34 bands.

Example screenshots:

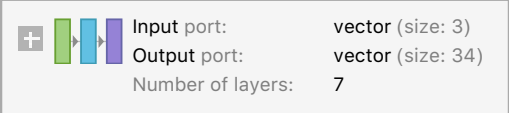


HOME-MADE HYPERSPECTRAL IMAGING

TRAINING A NEURAL NET

100 samples are used for training a net to transform 3 values into 34:

```
In[ ]:= ci = ReadList[
  "/Users/markon/nn_src/SPECTRUM/spec100.in", Number, RecordLists → True];
co = ReadList["/Users/markon/nn_src/SPECTRUM/spec100.out",
  Number, RecordLists → True];
trn = Flatten@Table[ci[[i]] → co[[i]], {i, 1, 100}];
nt = NetChain[{3, Tanh, 6, Tanh, 12, Tanh, 34}, "Input" → 3];
nt = NetInitialize[nt];
nttrn = NetTrain[nt, trn]
```

Out[]:= NetChain[]

Input port: vector (size: 3)
 Output port: vector (size: 34)
 Number of layers: 7

HOME-MADE HYPERSPECTRAL IMAGING

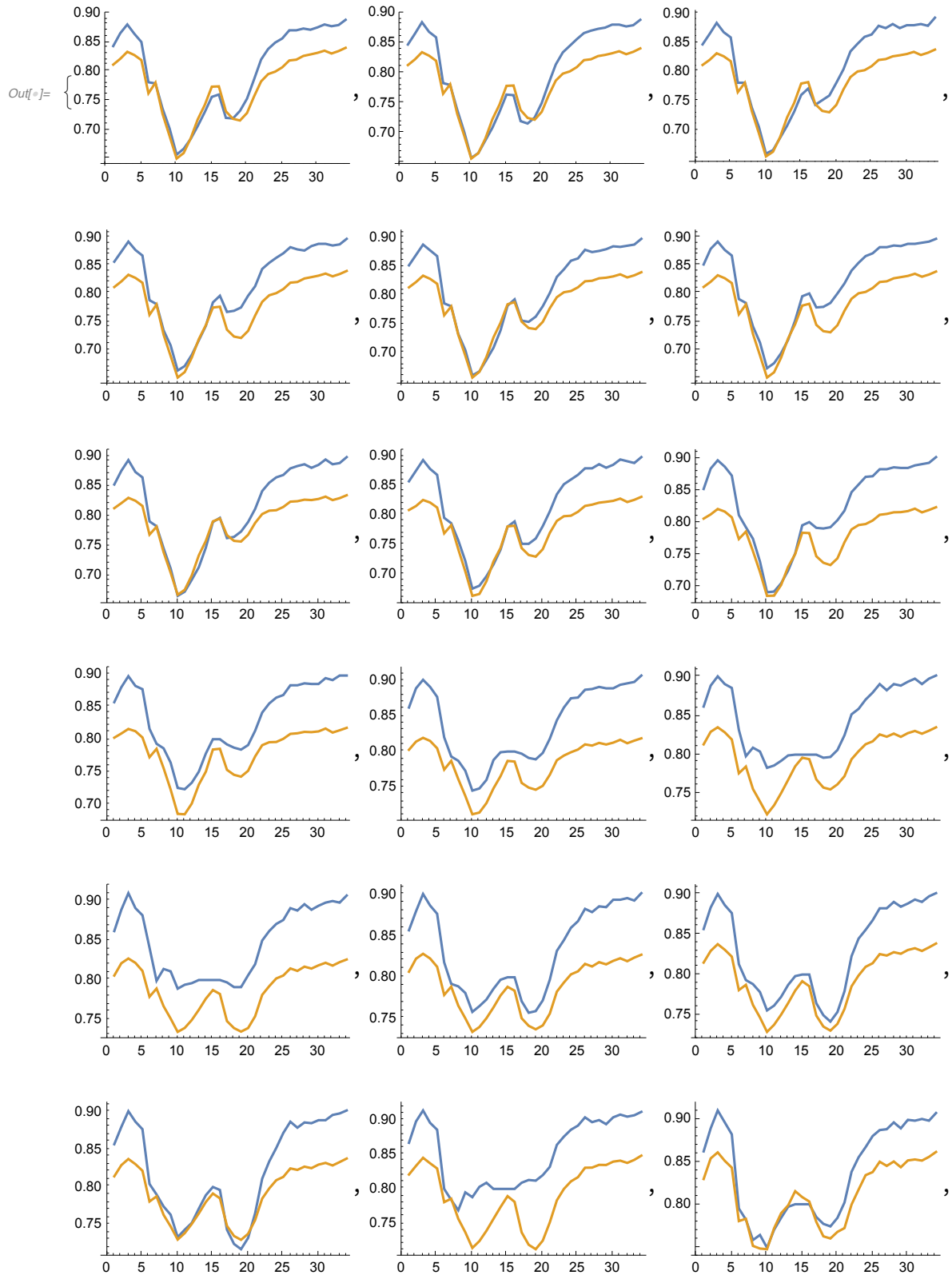
TESTING THE NEURAL NET

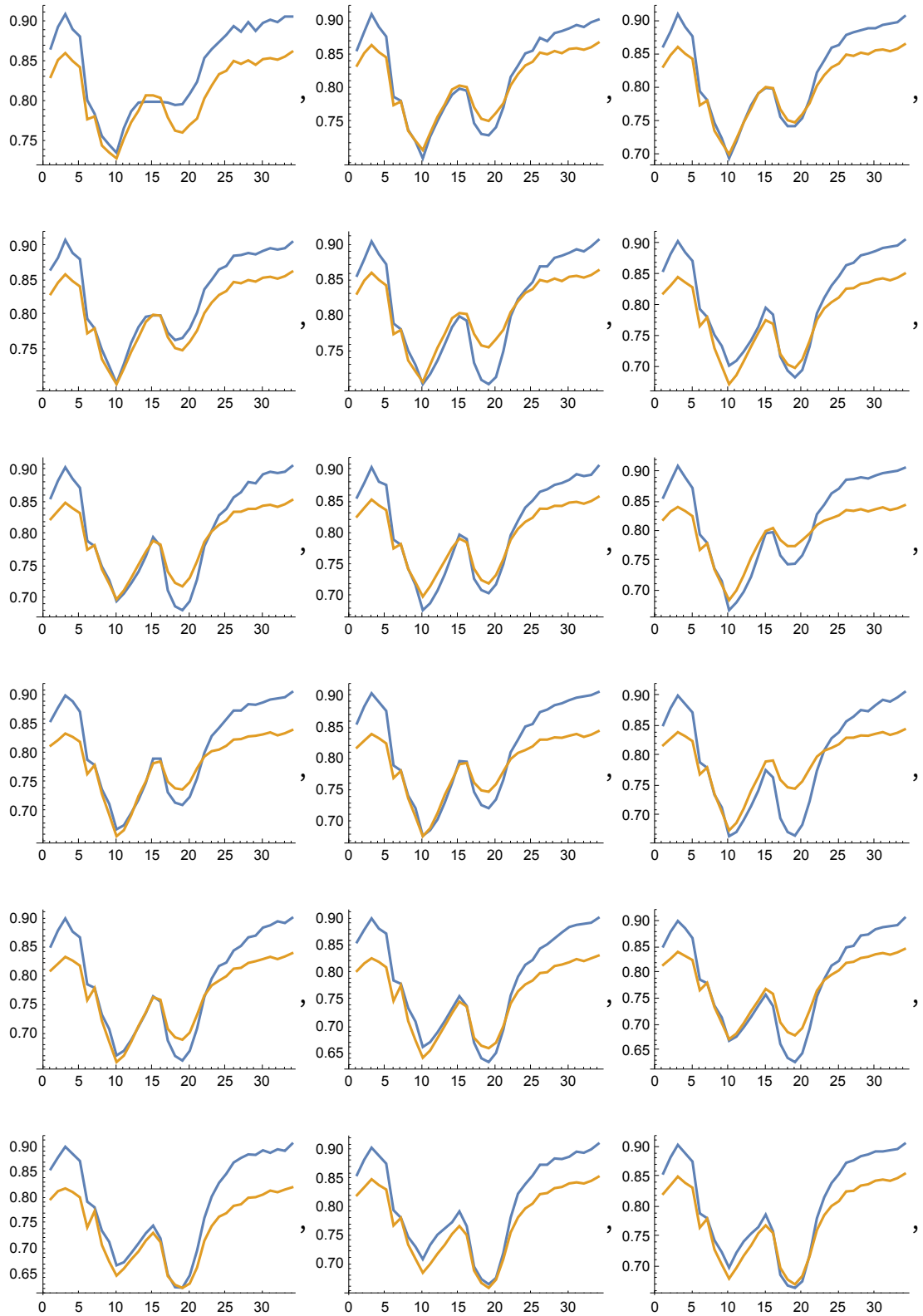
66 samples are used for testing the recall:

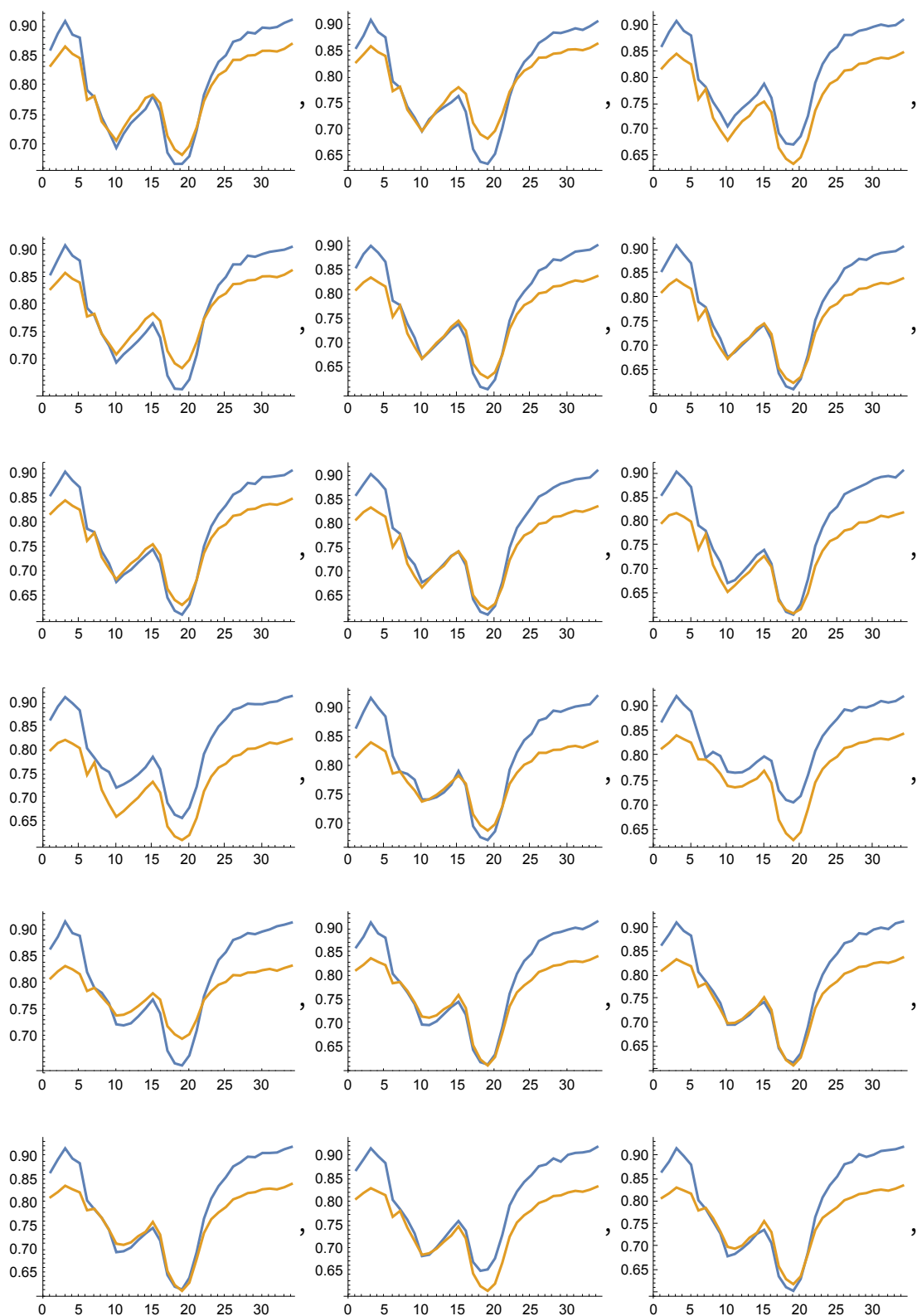
```
cti = ReadList[
  "/Users/markon/nn_src/SPECTRUM/spec66.in", Number, RecordLists → True];
cto = ReadList["/Users/markon/nn_src/SPECTRUM/spec66.out",
  Number, RecordLists → True];
ctot = nttrn[cti];
Table[ListPlot[{cto[[i]], ctot[[i]]}, Joined → True], {i, 1, 66}]
```

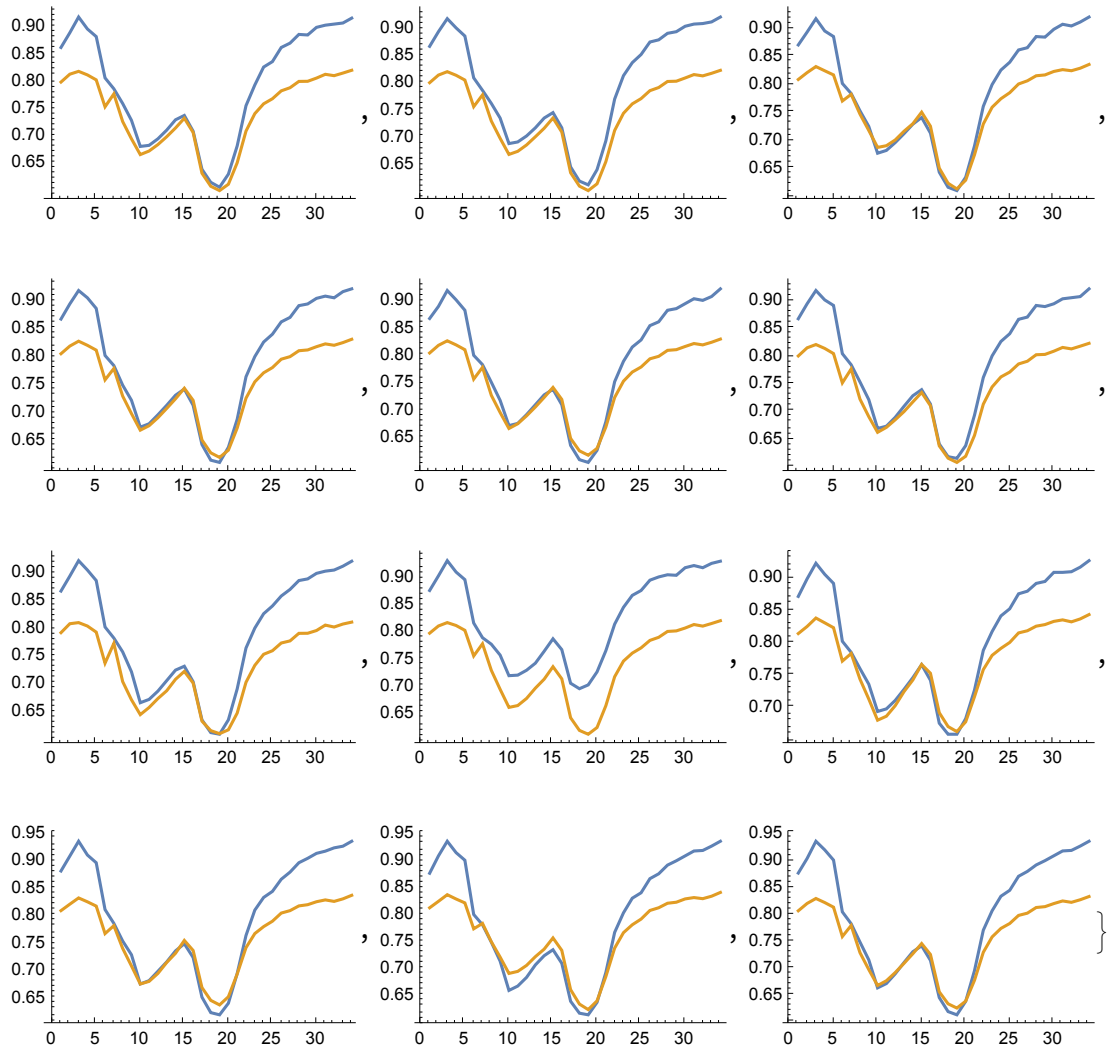
HOME-MADE HYPERSPECTRAL IMAGING

THE RESULTS: COMPARE MEASURED AND RECONSTRUCTED SPECTRUM









RGB IMAGE INTO SPECTRUM

Read in an RGB image

```
In[ ]:= pic = Import["/Users/markon/Pictures/Photos  
Library.photoslibrary/Masters/2018//05/27/20180527-083339/L1060749.JPG"];  
picSmall = ImageResize[pic, 480.]
```

Out[]:=



```
In[ ]:= picData = ImageData[picSmall]; Dimensions[picData]
```

Out[]:= {321, 480, 3}

RGB IMAGE INTO SPECTRUM

Create an image data sequence

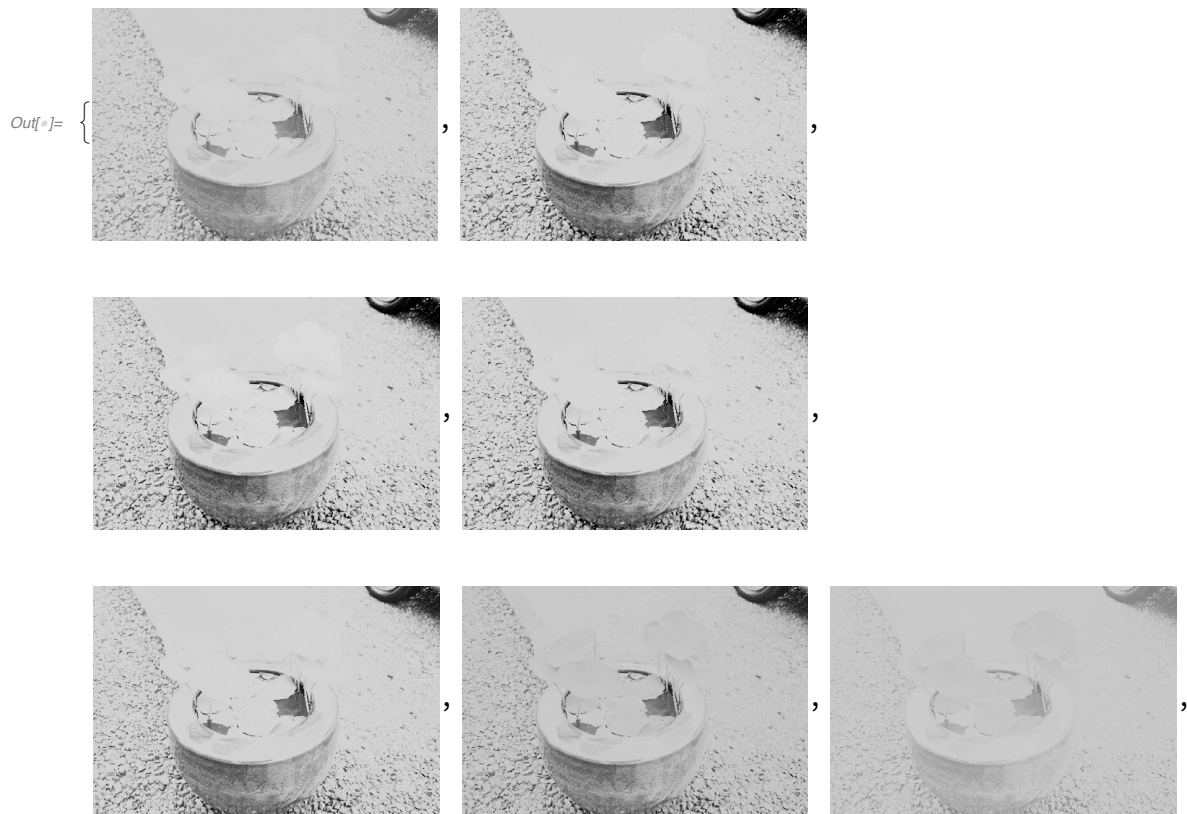
```
In[ ]:= specData = Table[1, {k, 1, 34}, {i, 1, 321}, {j, 1, 480}];
```

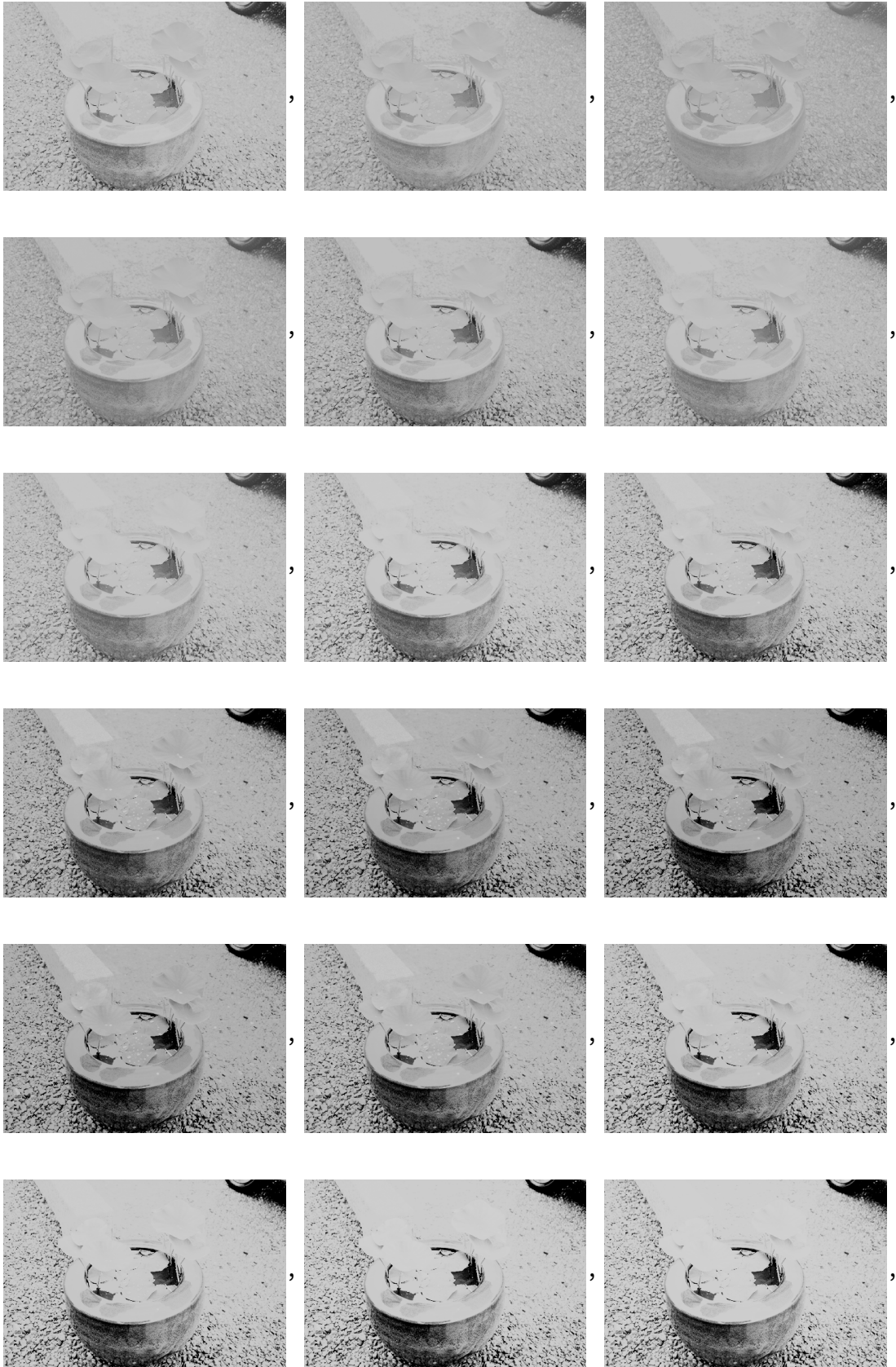
```
In[ ]:= p = picData[[1, 1]]; t = nttrn[p]
```

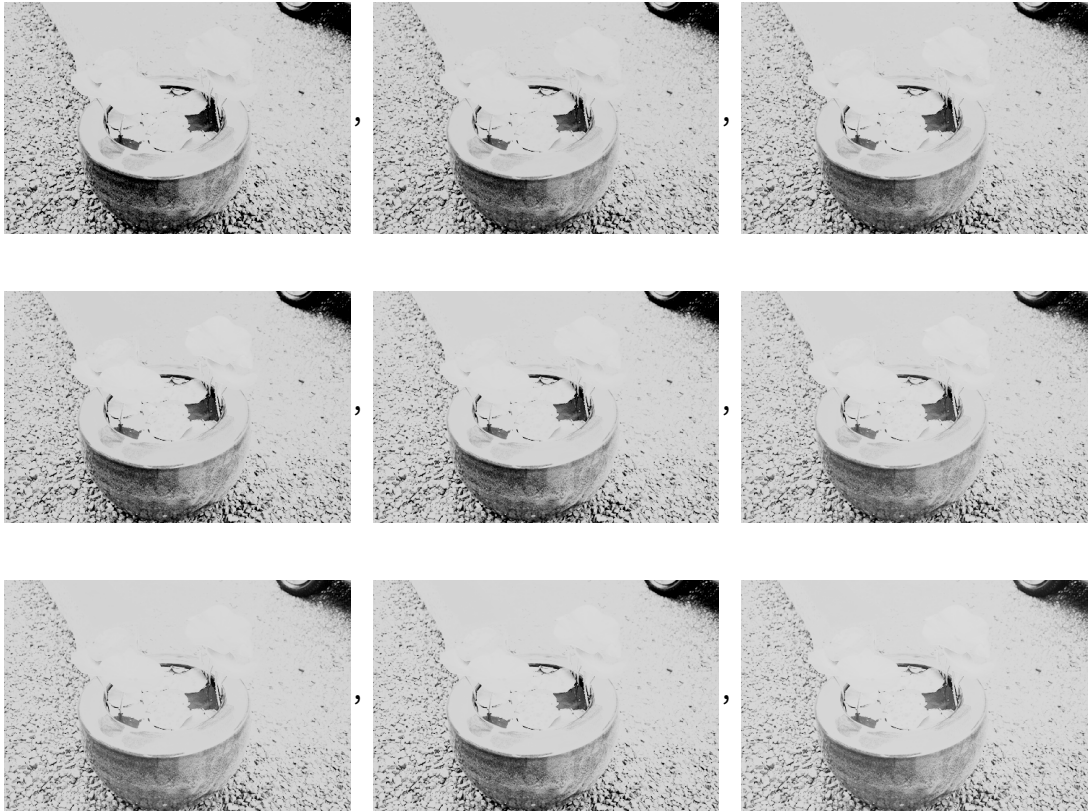
```
Out[ ]:= {0.812482, 0.826067, 0.836046, 0.830167, 0.821901, 0.770912,  
0.785235, 0.738331, 0.70819, 0.679561, 0.690036, 0.708348, 0.725494,  
0.743871, 0.766098, 0.75223, 0.696074, 0.674481, 0.667395, 0.681003,  
0.716271, 0.757231, 0.778221, 0.791881, 0.801271, 0.814895, 0.816738,  
0.823915, 0.82325, 0.829171, 0.832146, 0.830173, 0.834025, 0.840267}
```

```
In[ ]:= For[i = 1, i ≤ 321, i++,  
For[j = 1, j ≤ 480, j++,  
p = picData[[i, j]];  
t = nttrn[p];  
For[k = 1, k ≤ 34, k++,  
specData[[k, i, j]] = t[[k]]];]]
```

```
In[ ]:= Table[Image[specData[[k]]], {k, 1, 34}]
```







HOME-MADE HYPERSPECTRAL IMAGING

ONE SPECTRAL BAND

Notice the structure on the ceramic bowl that was hidden in RGB



HOME-MADE HYPERSPECTRAL IMAGING

ONE SPECTRAL BAND

Notice the structure on the ceramic bowl that was hidden in RGB



TODO

BETTER TRAINING, CALIBRATION AND TESTING

We need many color patches showing genuine spectral variation.

After training and validation, we need to compare with commercial spectroradiometers or at least hyperspectral imaging systems.

Finally, we need to test on actual images (plants etc.)

TRANSFER THE NEURAL NET TO SMARTPHONE

The system becomes useful when smartphone users can just install the app, point the camera on the scene, and scroll through the pseudo-spectral images.