Status of Code

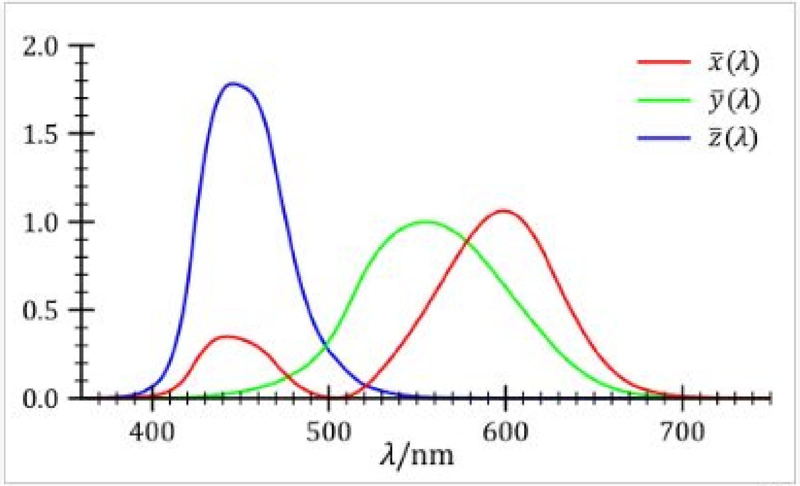
Bright & Sai

Previous Code

- Runs “genetic algorithm” for 40 generations based on priorities set by user
  - RGB based color algorithm was not matching LBNL’s color output
  - Used bulk silver nk, silver thickness - 3.5 nm constant -> not matching VLT and TSER
  - TSER calculated from 1 - T - 4% constant
  - Difficult to quickly ‘tune’ and make compromises between many variables (VLT, TSER, color)
- RGB based color algorithm
  - Weighted T/R by RGB color matching functions
  - Did not account for lighting source

New Color Algorithm

- CIE (International Commission on Illumination) XYZ -> L a b instead of RGB
- CIEXYZ: Y luminance, X Z color



The CIE standard observer colour matching functions.

The tristimulus values for a colour with a spectral radiance $L_{e,\Omega,\lambda}$ are given in terms of the standard observer by:

$$X = \int_{380}^{780} L_{e,\Omega,\lambda}(\lambda)\,\overline{x}(\lambda)\,d\lambda,$$

$$Y = \int_{380}^{780} L_{e,\Omega,\lambda}(\lambda)\,\overline{y}(\lambda)\,d\lambda,$$

$$Z = \int_{380}^{780} L_{e,\Omega,\lambda}(\lambda)\,\overline{z}(\lambda)\,d\lambda.$$

where λ is the wavelength of the equivalent monochromatic light (measured in nanometers).

New Color Algorithm -- LBNL

- Added weighting by “y” (luminance) function
- Changed from ASTM G173 to CIE D65 (Standard Illuminant)
- Switched from XYZ to LAB

$$X = \int_{.38}^{.78} P(\lambda)S(\lambda)\bar{x}_{10}(\lambda)\,d\lambda \Big/ \int_{.38}^{.78} S(\lambda)\bar{y}_{10}(\lambda)\,d\lambda$$

$$Y = \int_{.38}^{.78} P(\lambda)S(\lambda)\bar{y}_{10}(\lambda)\,d\lambda \Big/ \int_{.38}^{.78} S(\lambda)\bar{y}_{10}(\lambda)\,d\lambda$$

$$Z = \int_{.38}^{.78} P(\lambda)S(\lambda)\bar{z}_{10}(\lambda)\,d\lambda \Big/ \int_{.38}^{.78} S(\lambda)\bar{y}_{10}(\lambda)\,d\lambda$$

$$L^* = 116(Y/Y_n)^{1/3} - 16 \quad : Y/Y_n > 0.008856$$

$$L^* = 903.3(Y/Y_n) \quad : Y/Y_n \le 0.008856$$

$$a^* = 500\left[f(X/X_n) - f(Y/Y_n)\right]$$

$$b^* = 200\left[f(Y/Y_n) - f(Z/Z_n)\right]$$

where:

$$f(R) = (R)^{1/3} \quad R > 0.008856 \qquad : \; R = X/X_n,\; Y/Y_n,\text{ or } Z/Z_n \,.$$

and $\quad f(R) = 7.787(R) + 16/116 \quad R \le 0.00856$.

GenerateColor.py

```
for i in range(3):
        T_xyz[i] = np.trapz(T_interp * S_interp * xyz_sens[:,i], xyzwl) / float(np.trapz(S_interp * xyz_sens[:,1], xyzwl))
        R_xyz[i] = np.trapz(R_interp * S_interp * xyz_sens[:,i], xyzwl) / float(np.trapz(S_interp * xyz_sens[:,1], xyzwl))


X,Y,Z = T_xyz[:]
Xn, Yn, Zn = 95.047, 100.000, 108.883
L = 116*f(Y/Yn)-16
a = 500*(f(X/Xn)-f(Y/Yn)) + 0.4124  # this is how much "a" is off by, when we give 100% T or R -- Bright
b = 200*(f(Y/Yn)-f(Z/Zn)) - 0.9589 # this is how much "a" is off by, when we give 100% T or R -- Bright
```

- Interpolation and numerical integration (trapezoidal rule)
- Accounted for offset of +0.4124 in “a” and -0.9589 in “b”
    - Likely due to interpolation & numerical integration, outdated/inexact X,Y,Z color matching function

Color Results

- Using GenerateColor.py -- produced: L a b of SCC #10: (94.14, -2.46, -0.16) most color neutral to-date
- Error due to linear (2D) approximation of 3D vectors (LAB)

	Error SCC #10	Error newest Double Stack	Error SCC #5
L	0	-0.004	-0.012
a	-0.019	-0.047	-0.052
b	0.0513	0.1097	0.1456

New TSER Calculation

- LBNL's

The SHGC is a function of the total solar transmittance of the glazing system, see Section 7.2.1, the inward flowing fraction of absorbed solar energy and the absorptance of each layer, see Section 7.2.2:

$$SHGC_C = T_{1,N}^{sol} + \sum_{j=1}^{N} N_j A_j^{sol} \qquad 5.2a$$

Where N_j, the inward flowing fraction of absorbed solar energy is defined as the sum of the resistances up to a given node divided by the sum of all the resistances.

$$N_j = \sum_{m=1}^{j-1} R_m \Big/ \sum_{j=1}^{N} R_j \qquad 5.2b$$

Adapting to LBNL’s TSER Calculation

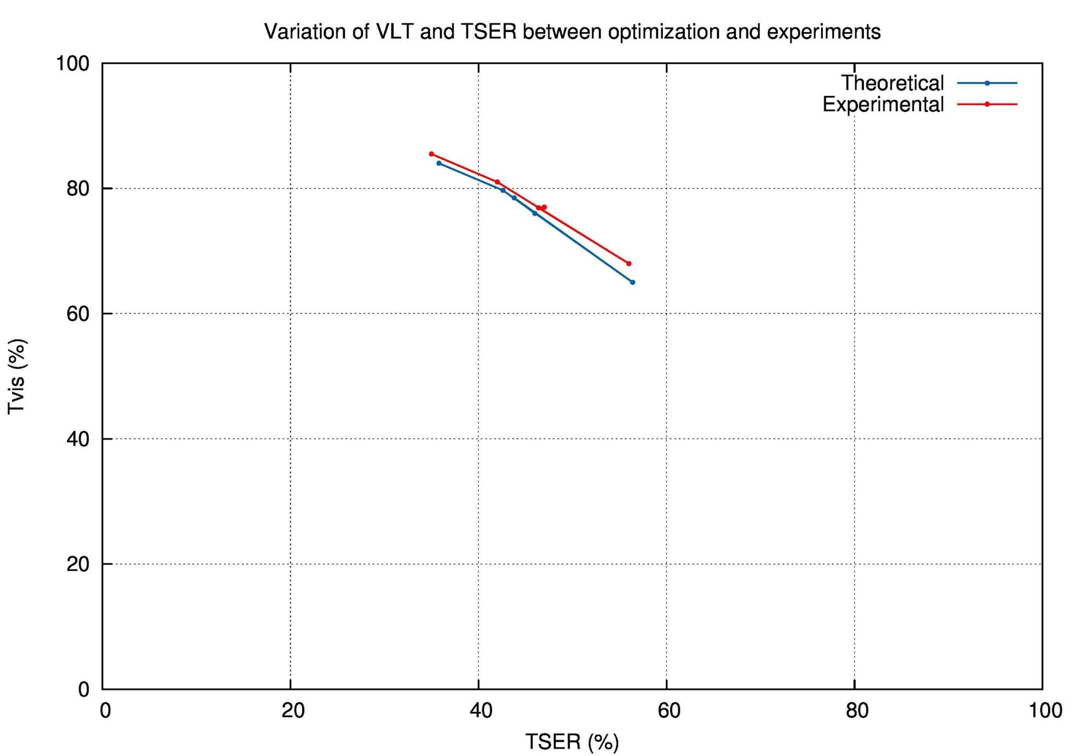
- Previously (by Remy:)

```
#TSER = 1 - sum(MyStack.T[index_lowerUV:index_upper]*sol_array)/(sum(sol_array)) - 0.04
```

- Now in UserInput.py:
  - TSER = sum R (weighted by solar) + 0.85 * sum A (weighted by solar)
  - Assuming 85% of absorbed is emitted away

Silver Refractive Index & TSER

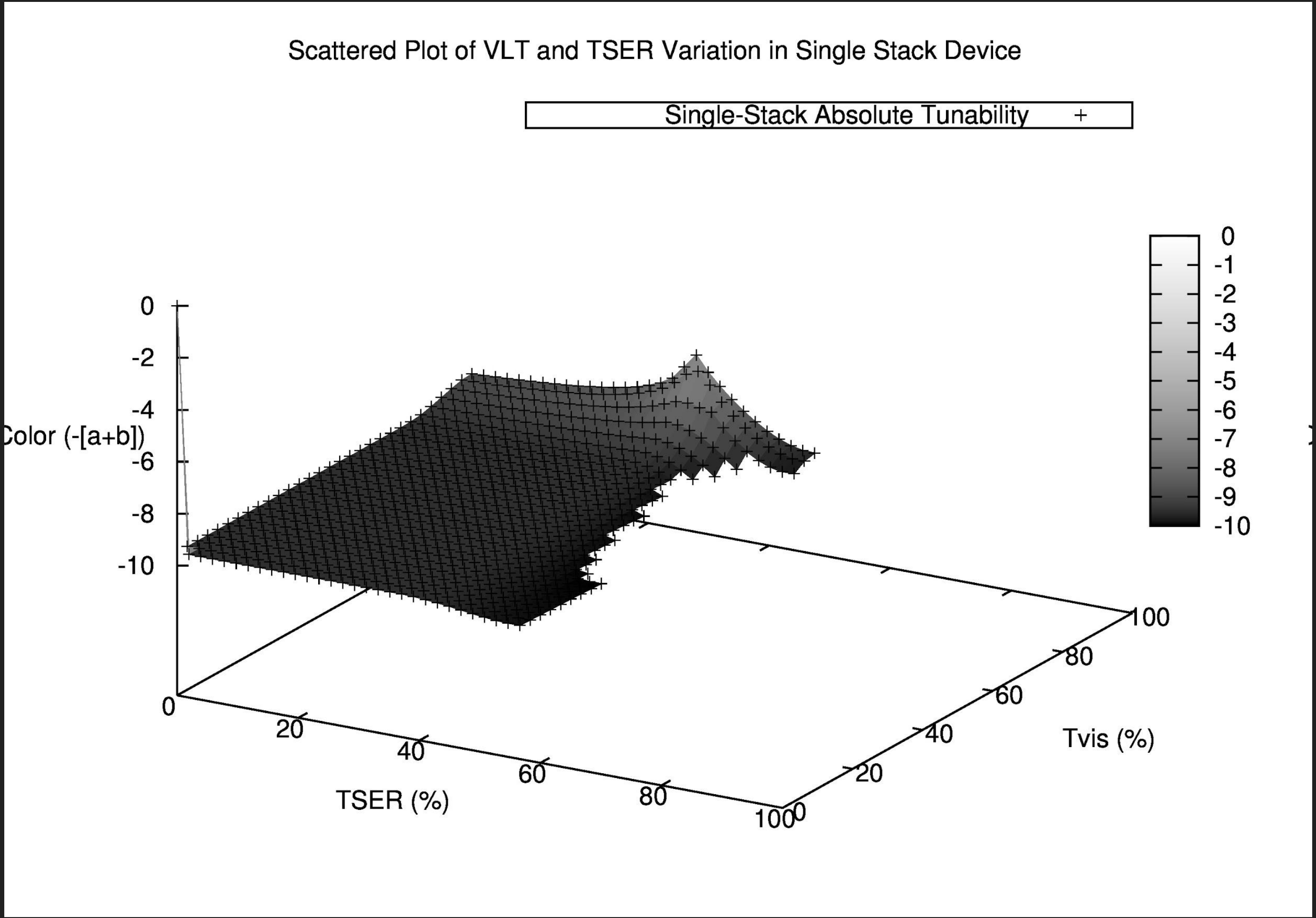
- Closest available Nishanth experimental nk data:
  - @ 8.75 nm silver (sputtered with N2)
  - @ 10.2 nm silver (sputtered with N2)
- Used 8.75 nm data extrapolated with 10.2 nm



GenerateAll.py

- Understanding of ‘tunability’ from pov of ‘optimization’ to ‘user choice’
- For thickness combinations:
  - AlNH seed in [3, 15]
  - Ag in [8.5, 14]
  - AlNH in [40, 55]
- Full outputs of VLT, TSER, L, a, b -- can be expressed in 3D graphical form

Final Results of All Parameters



Summary

- Effective changes:
  - (GenerateColor.py) Used CIE Standard Illuminant D65 for more accurate lighting conditions instead of ASTM G173
  - (GenerateColor.py) Used CIEXYZ-> CIELAB from LBNL to correctly predict color
  - (UserInput.py) Used extrapolated 8.75 nm Ag nk data to give accurate silver thickness
  - (UserInput.py) Used LBNL TSER calculation to closely predict TSER
  - (GenerateAll.py) Used same algorithm as UserInput, repeated over all possible combination of thicknesses to generate all possible single stack results
- What we have now:
  - Correct color prediction
  - More precise silver thickness and TSER
  - A database of all available performances of single stack to ‘pick and choose from’