INTRINSIC IMAGES IN THE WILD

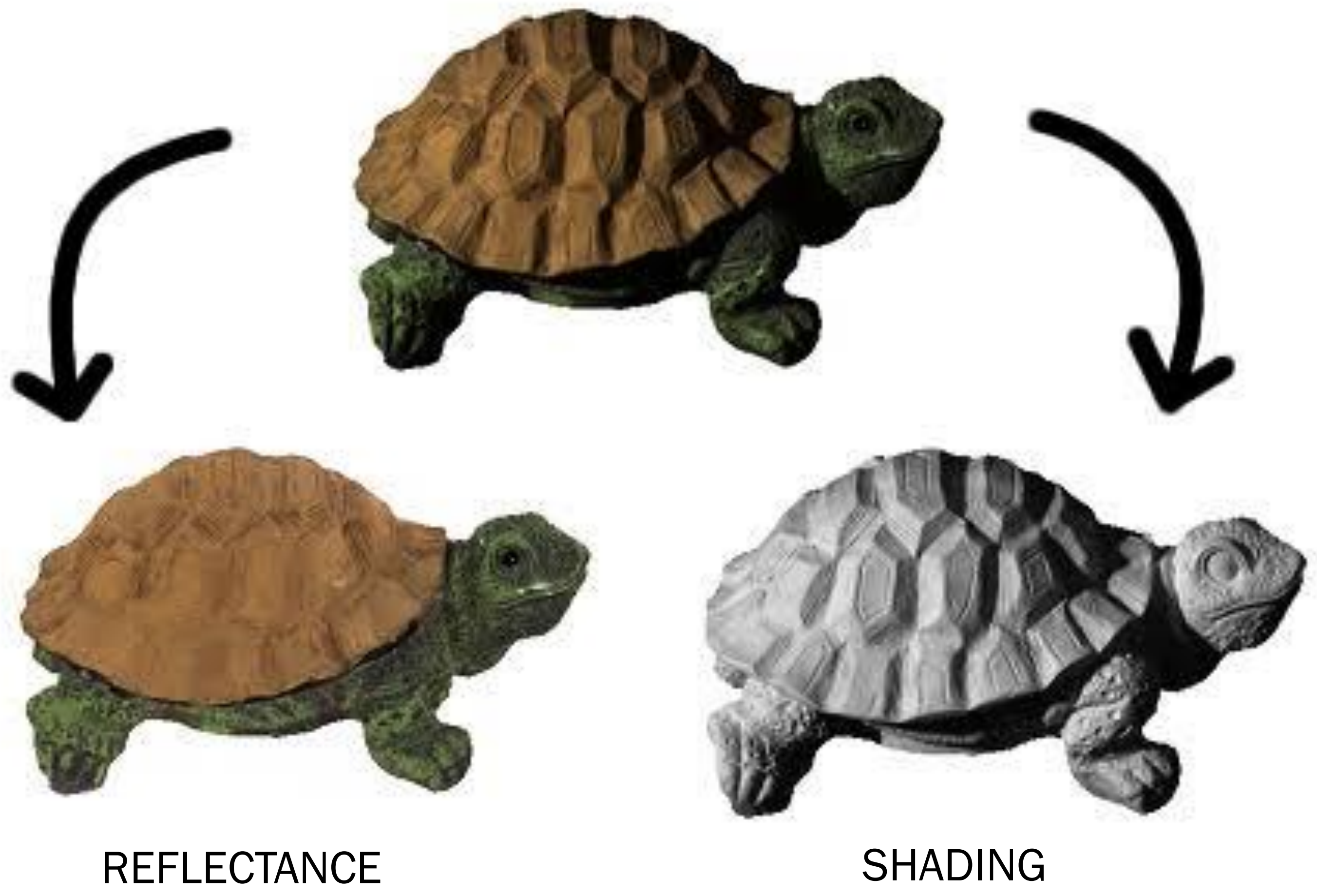
- THE FAST AND THE CURIOUS

- SAUMYA SHAH (20171193)
- ANIKET MOHANTY (20171150)
- SOUMYASIS GUN (20171002)
- MENTOR TA : KARANDEEP SINGH JUNEJA

- REPO : HTTPS://GITHUB.COM/DEFUS3R/INTRINSIC-IMAGE-DECOMPOSITION

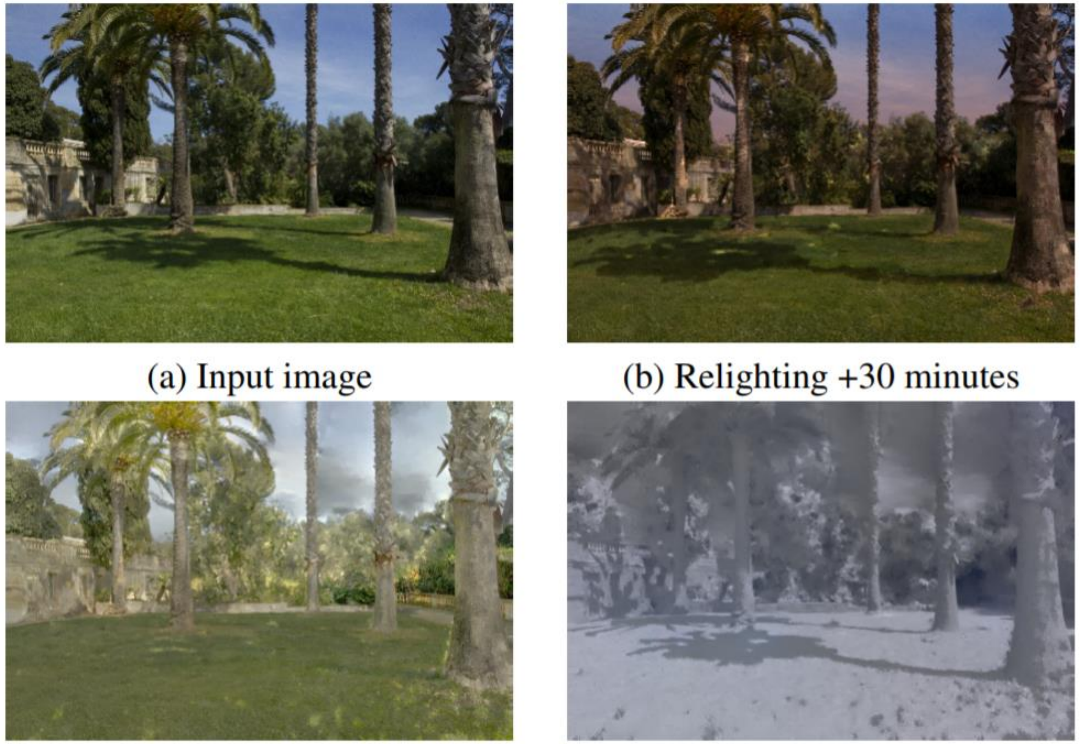
AIM

- Intrinsic images are a decomposition defined by image reflectance and shading.
- Answering questions like :
  - What effect is the lighting having, irrespective of surface materials?
  - What is the surface reflectance, irrespective of lighting?



MOTIVATION

- Intrinsic images along with cast shadows information, can be used for image relighting.
- It can also be used for texture composition that preserves the lighting information.

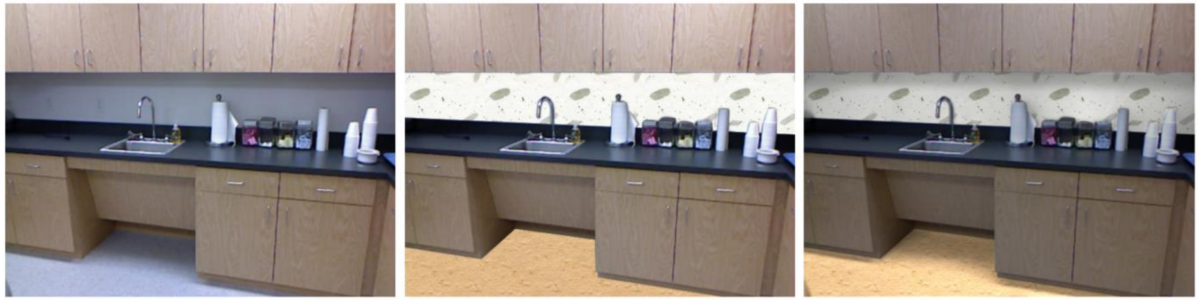


(a) Input image (b) Relighting +30 minutes

(c) Reflectance (d) Shading

Image Relighting

Texture composition



(a) Original image (b) Naive copy & paste (c) Our method

INTRINSIC IMAGE DECOMPOSITION

- Intrinsic image decomposition gives output as reflectance (R) and shading (S).
- $\boxed{\text{Very under-constrained problem.}}$
- Infinitely many possible reflectance and shading layers that multiply to explain an input image.
- Need to find reflectance layer R* and shading layer S* that is most likely under the probability distribution p.

$$R^*, S^* = \underset{R,S}{argmin}\ p(R, S \mid I)$$

such that $I_i^c = R_i^c \cdot S_i^c$ for every image pixel i per channel c.

MOTIVATION FOR CONSTRAINTS FROM PRIOR WORKS

Pixels that are nearby, and that have similar chromaticity or intensity, also have similar reflectance.

Reflectances are piecewise-constant

[Land and McCann 1971; Liao et al. 2013; Barron and Malik 2013b].

Reflectances are sampled from a sparse set

[Omer and Werman 2004; Gehler et al. 2011; Shen and Yeo 2011].

Certain shading values are a priori more likely than others

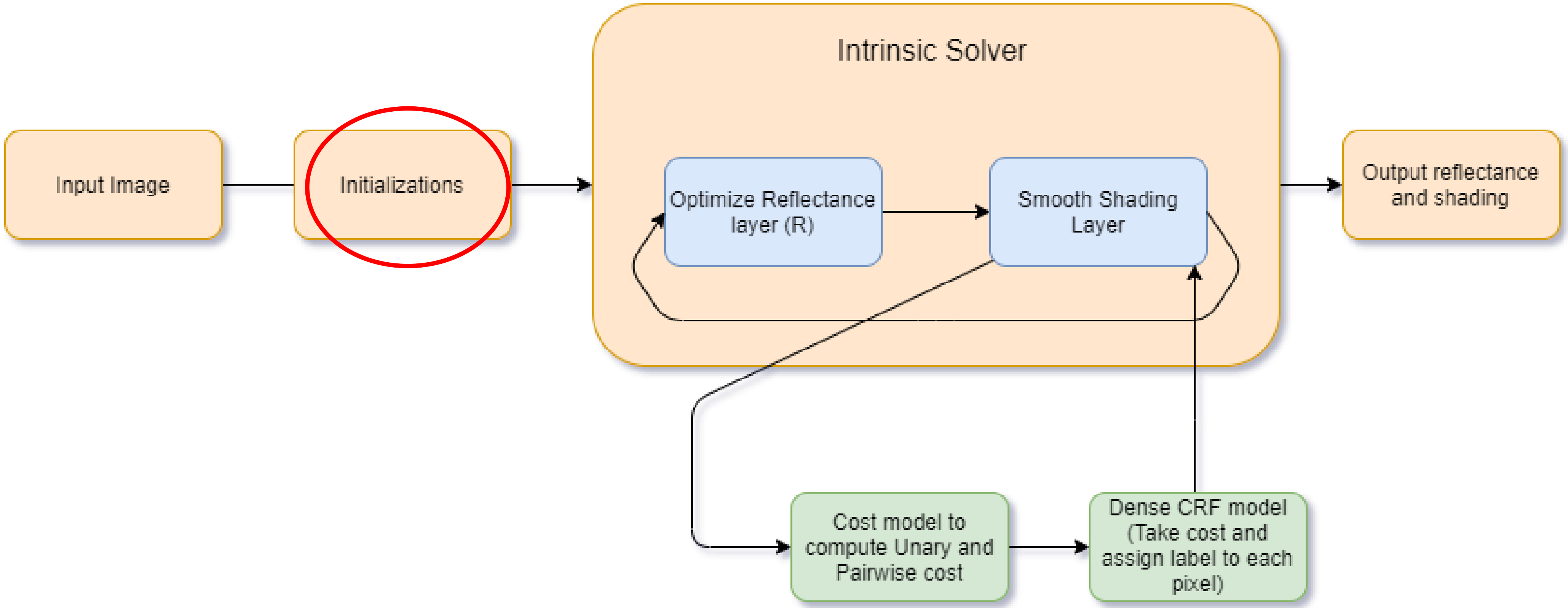
[Barron and Malik 2013b].

Neighboring pixels have similar shading

[Garces et al. 2012].

Shading is grayscale, or the same color as the light source.

ALGORITHM OVERVIEW



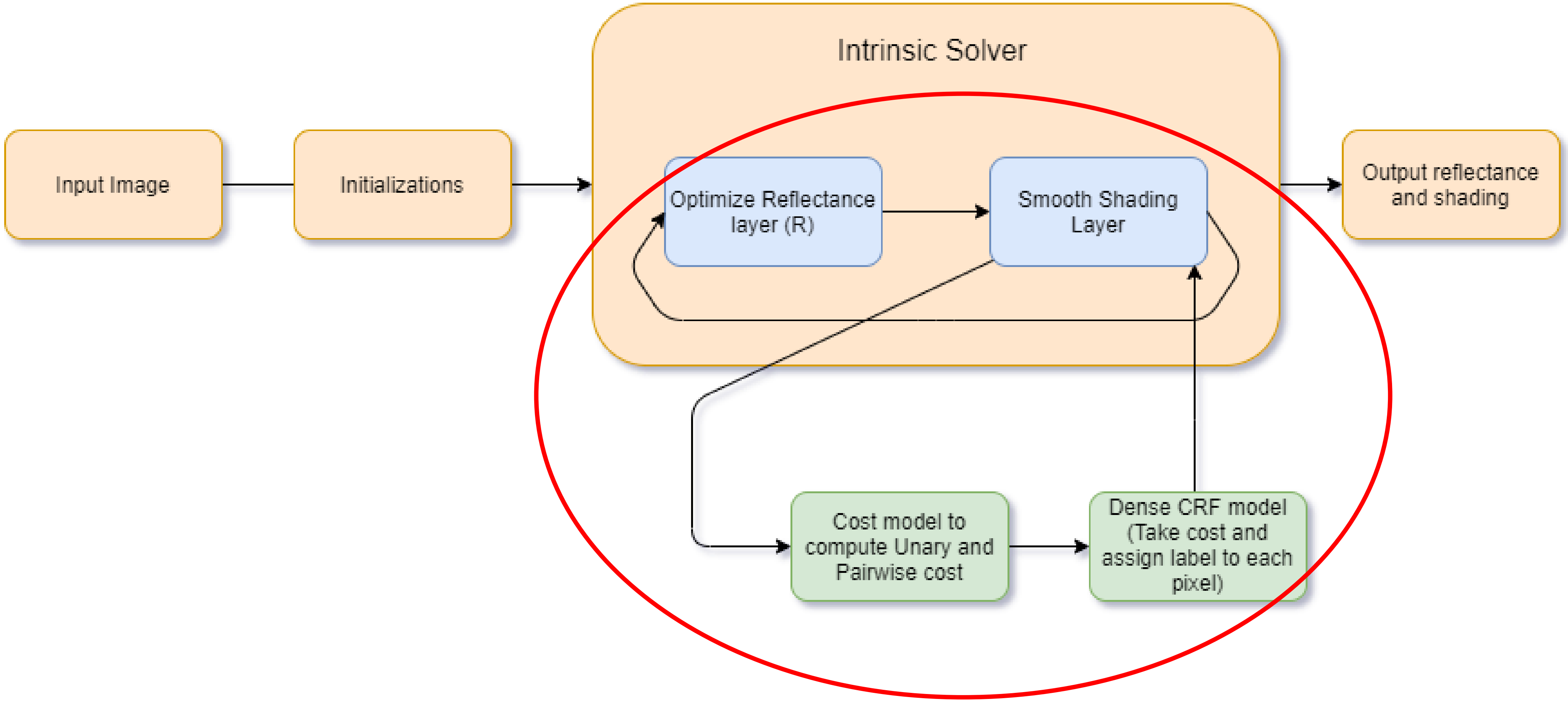
INITIALIZATION

Convert RGB image to IRG space (intensity, red chromaticity, green chromaticity)

Run k-means on this for 20 clusters and store the centers as recfllectances in R

Project image back to RGB space

ALGORITHM OVERVIEW



SOLVER STAGES

Stage 1:

Assign each pixel a reflectance selected from R

Stage 2:

Optimizing the set of reflectances R itself to improve shading intensity S.

STAGE 1 : FRAMING OBJECTIVE FUNCTION

- Optimal reflectance and shading are estimated using a minimization problem.

$$E(x) = \underbrace{w_p E_p(x)}_{\text{pairwise } \psi_{ij}} + \underbrace{w_s E_s(x) + w_l E_l(x)}_{\text{unary } \psi_i}$$

- Where x is input pixel and w_p, w_s, w_l are weights for the energy functions.
- Energy function consists of two types of costs:
  - Unary costs ψ_i for each pixel, where shading should be smooth and must avoid extreme values.
  - Pairwise costs $\psi_{i,j}$ where reflectance should be piecewise constant for all pairs of similar pixels.

HOW IS THE OPTIMIZATION DONE ?

- Fully connected conditional random fields are used to optimally give the pixels the labels of the reflectances that we stored in the set R after running k-means on the IRG image.
- This is a discrete labelling problem of maximizing $p(x \mid I)$.
- Or minimizing the log-likelihood.
- Krahenbuhl and Koltun proposed an algorithm for approximately minimizing a global objective function with a quadratic number of terms in linear time. This algorithm is popularly called DenseCRF.

$$p(x \mid \mathbf{I}) = \frac{1}{Z(\mathbf{I})} \exp\left\{ -\sum_i \psi_i(x_i) - \sum_{i<j} \psi_{ij}(x_i, x_j) \right\}$$

$$\psi_{ij}(x_i, x_j) = \sum_m \mu^{(m)}(x_i, x_j)\, k^{(m)}(\mathbf{f}_i - \mathbf{f}_j)$$

$p(x \mid \mathbf{I})$	probability of labels x given image $\mathbf{I}$
$Z(\mathbf{I})$	partition function (normalizes the distribution)
i, j	pixel indices
x_i	discrete label for pixel i
$\mathbf{f}_i$	feature vector for pixel i
$\psi_i(\cdot)$	unary cost function for pixel i
$\psi_{ij}(\cdot,\cdot)$	pairwise cost function for pixels i, j
$\mu^{(m)}(\cdot,\cdot)$	m^{th} (negative-semidefinite) label compatibility function
$k^{(m)}(\cdot)$	m^{th} (positive-definite) kernel function

REFLECTANCE ENERGY E_P

- $E_P = \sum_{i<j} \mu(x_i, x_j) \exp\left(\frac{-1}{2} \|f_i - f_j\|_2^2\right)$
- Where $\mu(x_i, x_j) = \|\log R(x_i) - \log R(x_j)\|_1$ is the label compatibility function.
- And f_i is the feature vector for pixel i:

$$\mathbf{f}_i = \left[\frac{p_i^x}{\theta_p\, d}, \frac{p_i^y}{\theta_p\, d}, \frac{\frac{1}{3}\sum_c I_i^c}{\theta_l}, \frac{I_i^r}{\theta_c \sum_c I_i^c}, \frac{I_i^g}{\theta_c \sum_c I_i^c}\right]$$

- For every pair of pixels in the image, if the two pixels are assigned a different reflectance, a cost proportional to the L^1 difference in this reflectance is paid and is Gaussian-weighted according to the distance between the pixels in our feature space.

SHADING SMOOTHNESS E_S

- $E_S\left(x^{(t)}\right) = \sum_i \left(\log S_i^{(t)} - \log \tilde{S}_j^{(t)}\right)^2$
- $\tilde{S}_i^{(t-1)} = \frac{1}{A_i} \sum_j S_j^{(t-1)} \exp\left(\frac{-1}{2} \left\| \frac{p_i - p_j}{\sigma_S^{(t)}} \right\|_2^2\right)$

  where A_i is a normalizing term.
- S_j is approximated by solving the model iteratively using the shading channel from the previous iteration; thus giving the functional form of an unary term as shown above.

REGULARIZATION TERM E_l

- $E_l(x) = \sum_i |S_i - \bar{S}|$
- In addition to encouraging smoothness of shading, we need to ensure that the optimizer does not choose extreme values of shading for too many pixels, so a penalty is added that pulls shading intensity S towards a constant.
- $\bar{S} = 0.5$ works best as suggested in the paper.

STAGE 2 : SHADING OPTIMIZATION

- In Stage 1, we obtained a hypothesis decomposition (R, S), as well as an indication of which pairs of points have the same reflectance (i and j have the same reflectance iff x_i = x_j). In Stage 2, we hold the label assignments x fixed and continuously optimize the set of reflectances R in order to improve shading intensity S.
- Inside an image region that is assigned a single reflectance label, shading is already constrained (since setting a value for either R or S fixes the other layer). Thus, if we hold the labels x fixed, we can only minimize shading discontinuities across reflectance boundaries, where $x_i \neq x_j$.
- We can update the set of reflectances by multiplying R by a vector of scalars r, which allows us to regularize the relative change r:

$$R^{(t)}(i) = r^{(t)}(i) \cdot R^{(t-1)}(i)$$

- We solve for r by minimizing shading discontinuities:

$$r^{(t)} = \underset{r}{argmin} \sum_{(i,j) \in B} \left| \log S_i - \log S_j \right|$$

$$\log S_i = \log\left(\sum_c I_i^c\right) - \log\left(\sum_c R^{(t-1),c}(x_i)\right) - \log r(x_i)$$

NOVELTY

1. Priors with colour information:
   - The priors for shading suggested by the paper uses only grayscale information for shading.
   - If the lighting isn't of white colour(or close to white) then the expected reflectance is not achieved.

Original Image

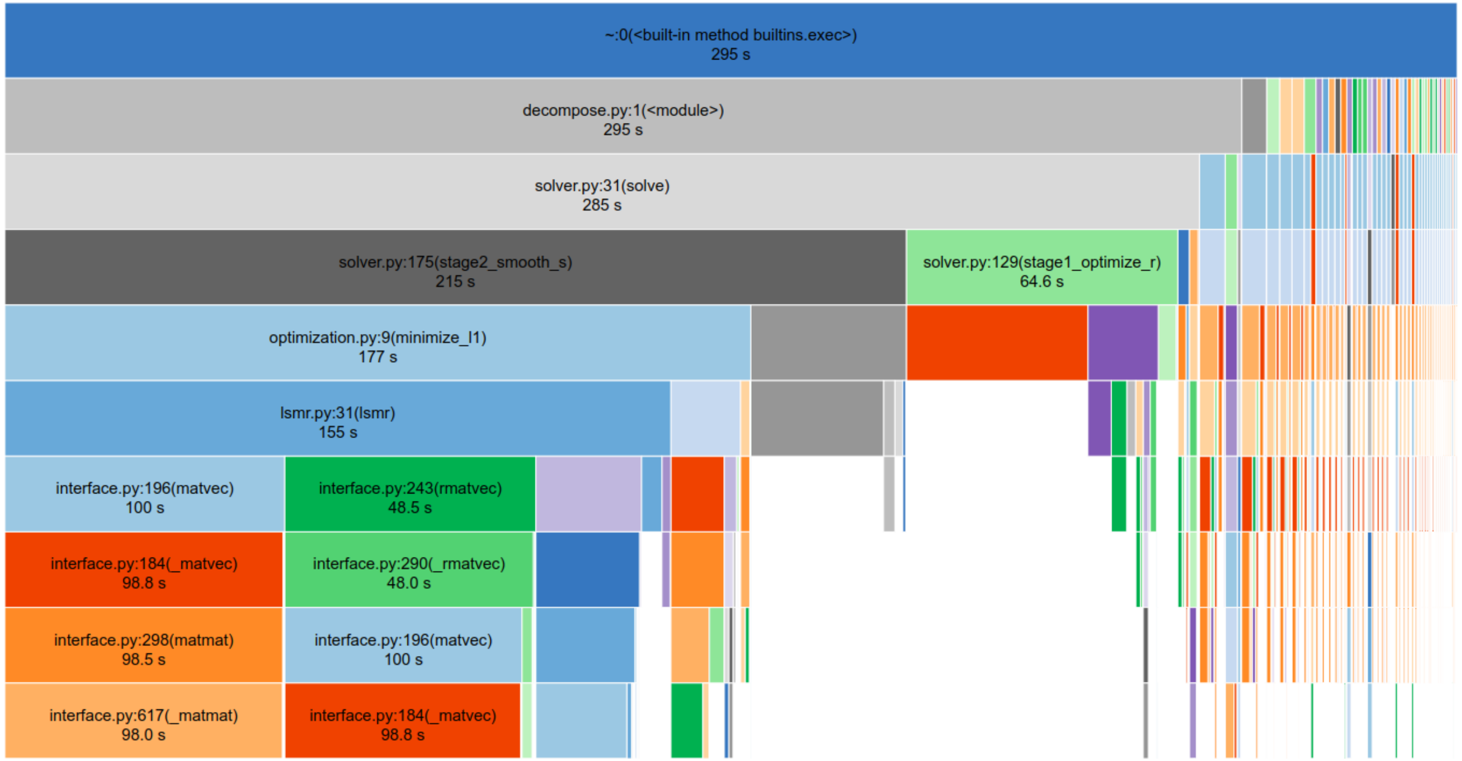
Shading Image

Achieved Reflectance

- We tried to incorporate RGB information of source by taking correlation of each channel with every other channel and finding independent probabilities; but failed at implementing it.

2. Tried to implement speed-up using numba:

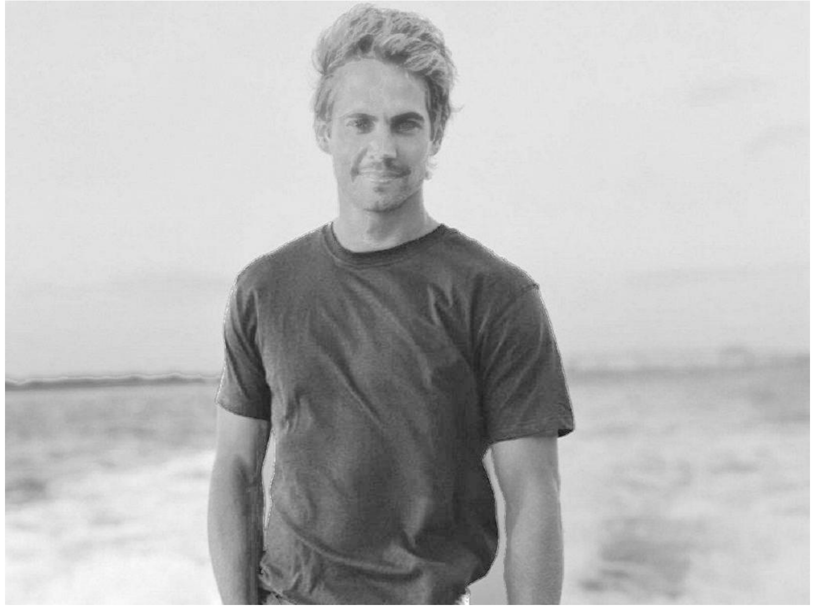
- This was done by viewing the memory stack of each file via snakeviz visualization.



Memory Stack

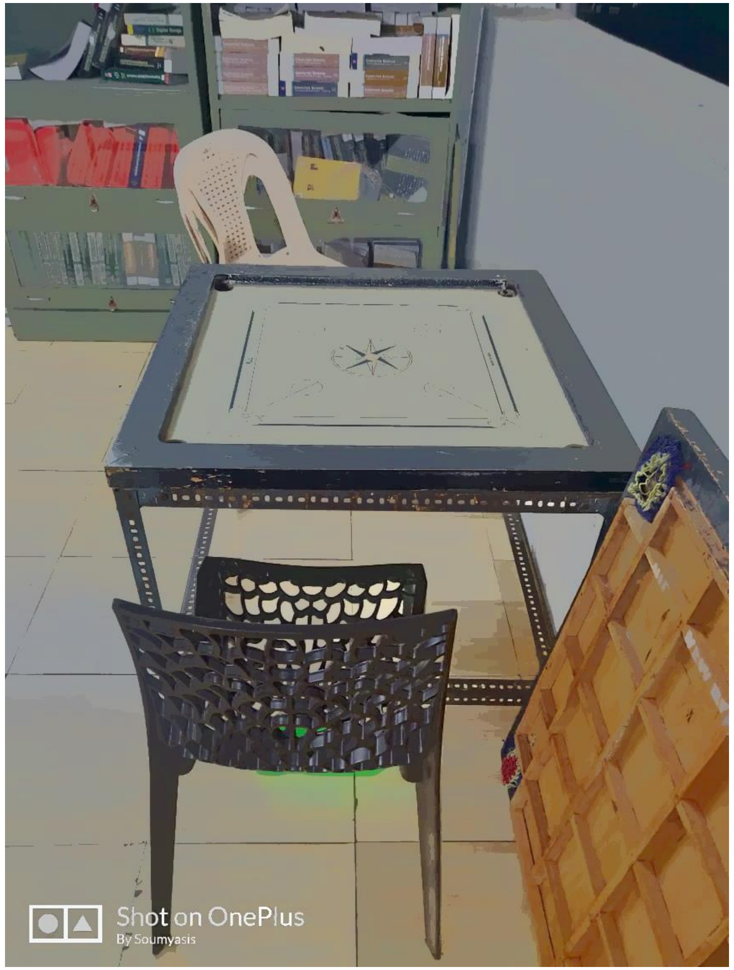
- This novelty couldn't be completed due to problems with jit class of numba and driver issues with CUDA.
- An additional way this speed up could be targeted is using multi-core CPU threading using LLVM. However due to time constraints we have put it as a future objective.

RESULTS



Original Image

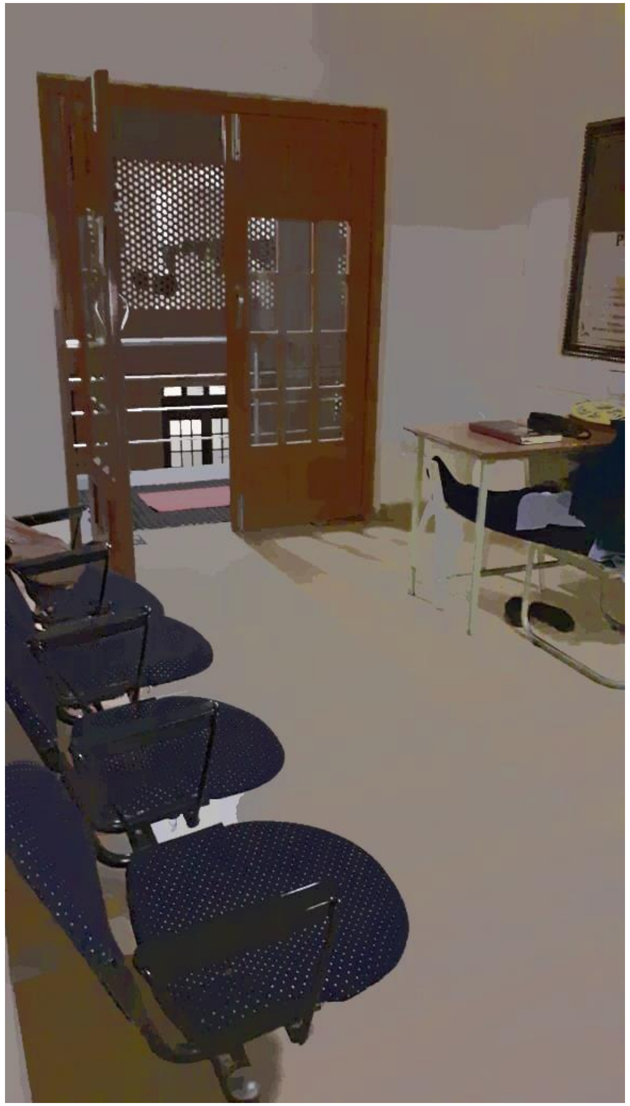
Reflectance Image



Shading Image

Original Image

Reflectance Image



Shading Image


WHEN DID YOU BECOME
AN EXPERT IN THE
THERMONUCLEAR ASTROPHYSICS?
LAST NIGHT

WORK DIVISION

Saumya :

Dataset Management

Histogram of probability density class.

Update parameters.

Obtain judgement parameters for opensource dataset.

Aniket :

L1 and L2 optimisation for minimizing shading discontinuities.

Created wrapper for DenseCRF [from Krahenbuhl].

Integrated cpp files.

Soumyasis :

Documentation.

Created function for image interface.

Algorithm initialization.

Repaired the energy function.

Combined work:

Saumya and Soumyasis :

- Built solver class

Aniket and Soumyasis :

- Energy function optimizer for stage 1

All together : Debugging

THANK YOU