TRANSFORM RECIPES FOR EFFICIENT CLOUD PHOTO ENHANCEMENT

Team - HYDROHOMIES
Mentor - Surendra Kumar Reddy
Link to Github

Abhishek Gangapuram
Kalyan Adithya
Jaiwanth Mandava
Sai Shashank Kalakonda

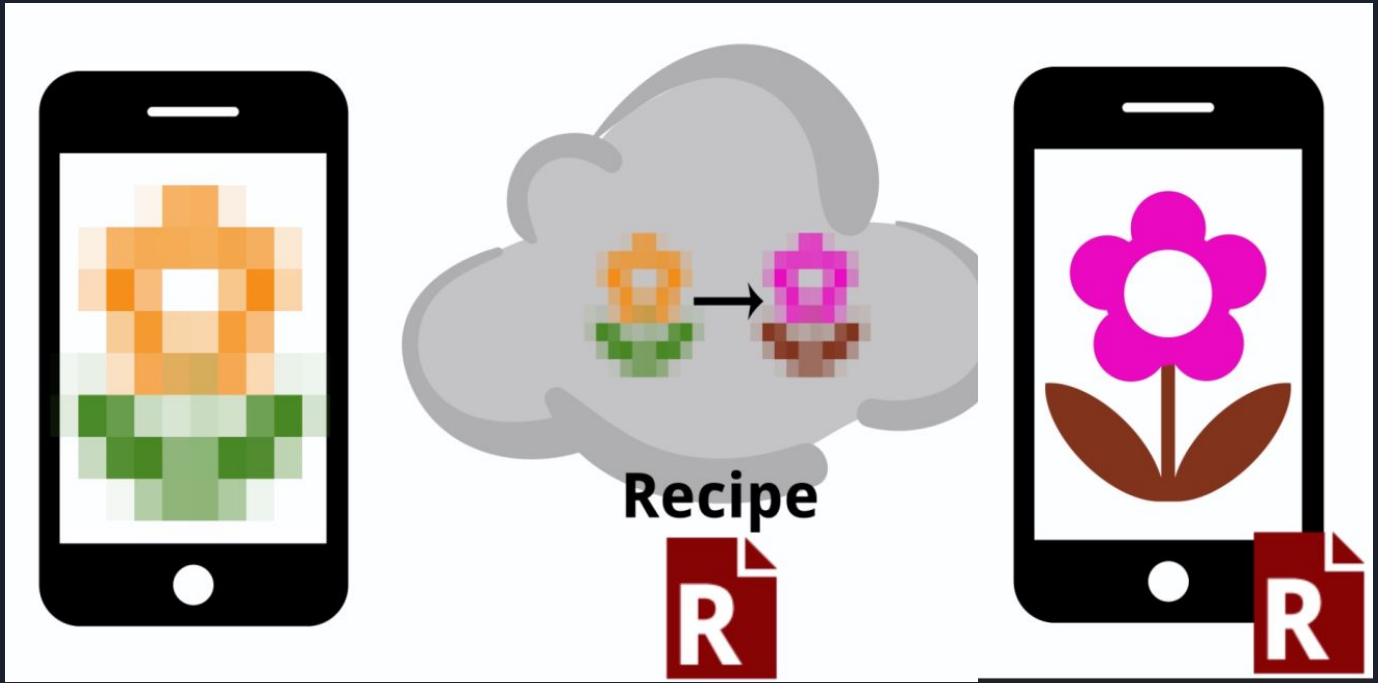
Main Goal

- To implement a pipeline that reduces the time and energy cost of uploading an image and downloading the output images after applying certain transformations.
- To use the similarities in input and output images when image enhancement techniques are applied.

Problem Definition

- Cloud computing is often thought as an ideal solution to enable complex algorithms on mobile devices; by exploiting remote resources, one expects to reduce running time and energy consumption. However, this ignores the cost of transferring data over the network, the overhead of which often negates the benefits of the cloud, especially for data-heavy image processing applications.
- The paper introduces a new image processing pipeline that reduces the amount of transmitted data.

Proposed Solution



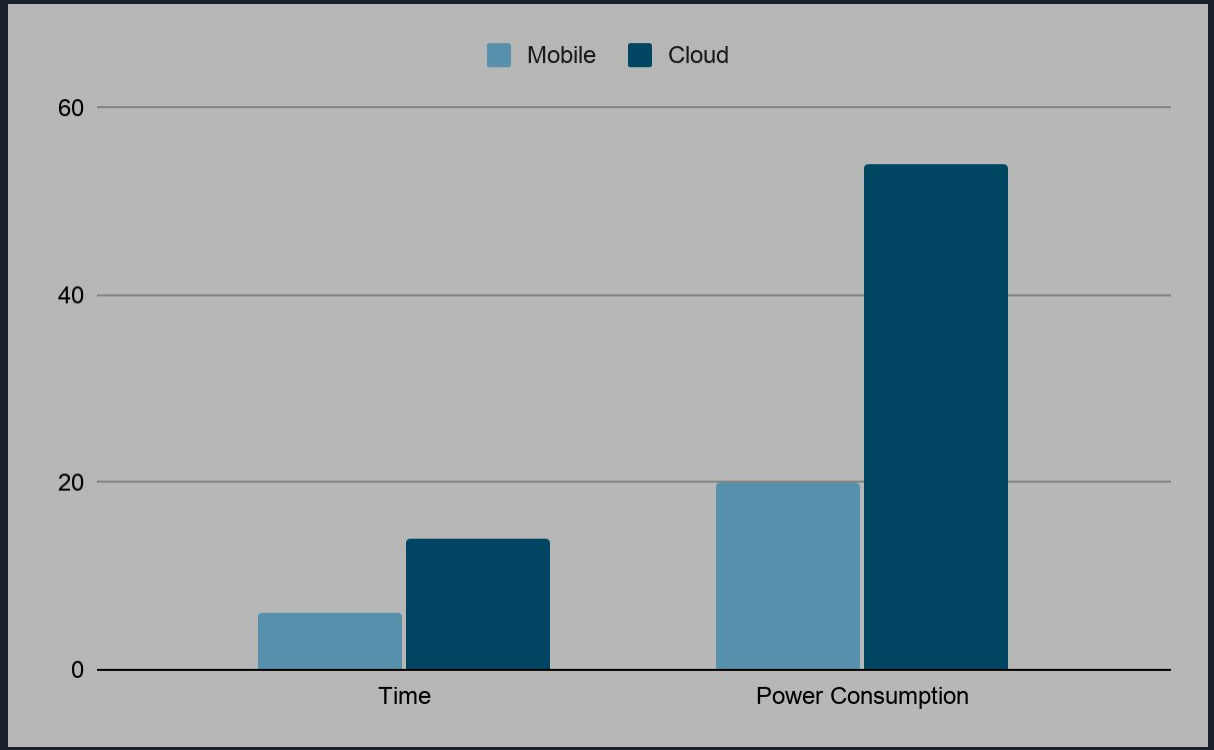
Naive Method

Upload

Transformation

Download

LOCAL vs NAIVE CLOUD



Naive Cloud vs Proposed Method

We clearly see that the power consumption and the total time that it took to process is quite high.

But the main reason to this was the data transfer from uploading the input and downloading the output.

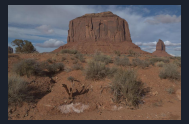
HOW TO DECREASE THE DATA EXCHANGE BETWEEN CLIENT AND SERVER??

Pipeline

18.5 MB

ORIGINAL INPUT

96 KB



Compressed Input

Histograms of top 3 layers of laplacian pyramids.

120 KB

15.3 MB

PROXY INPUT

CLIENT SIDE

SERVER SIDE

Compressing the Input

- Downsampling 2x to 8x
- Aggressive JPEG compression
- No high frequencies in the uploaded image
  - downsampling and JPEG compression
  - Could miss transform effects (e.g. sharpen)

SERVER SIDE

14.8 MB

PROXY INPUT

Transformation Applied

17.9 MB

PROXY OUTPUT

RECIPE

186 KB

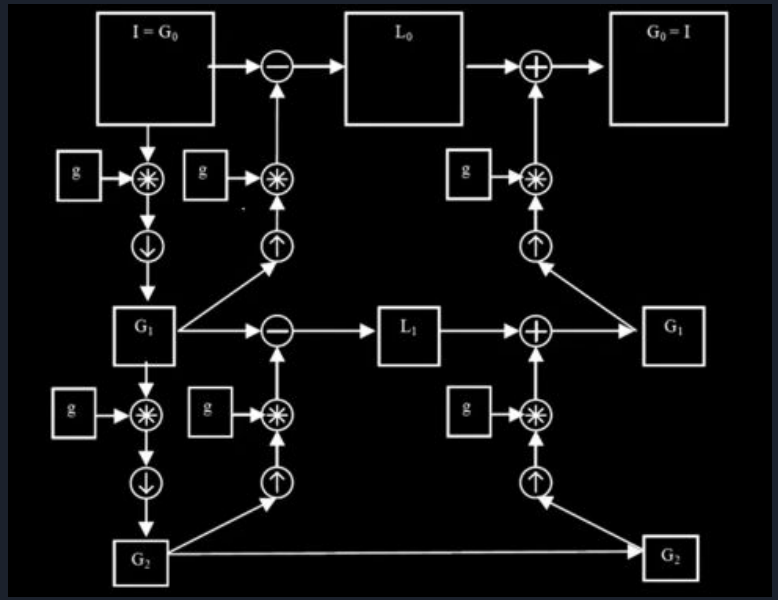
Recipe of 186 KB size, to be sent to client

What is RECIPE ?

- We exploit the pixel-wise similarities between the input and output image.
- We work on the images in YCbCr space, and separate Luminance Y from Chrominance (Cb, Cr) while capturing the transformation.
- To capture multi-scale effects, we decompose an image into a Laplacian stack

Laplacian Stack

- Laplacian Pyramid is formed by the difference of consecutive gaussians(after up-sampling the smaller image).



Math used in computation of Recipe

Residual Layer

$$R_c(p) = \frac{L_n[O_c](p) + 1}{L_n[I_c](p) + 1}$$

Combined High Frequency Data

$$H[I] = \sum_{\ell=0}^{n-1} L_\ell[I]$$

Chrominance Channels

$$\sum_{p \in \mathcal{B}} \| H[O_{CC}](p) - \mathbf{A}_c(\mathcal{B}) H[I](p) - \mathbf{b}_c(\mathcal{B}) \|^2$$

Luminance Channels

$$\sum_{p \in \mathcal{B}} \Big\| H[O_Y](p) - \mathbf{A}_Y(\mathcal{B}) H[I](p) - b_Y(\mathcal{B}) - \sum_{\ell=0}^{n-1} m_\ell(\mathcal{B}) L_\ell[I_Y](p) - \sum_{i=1}^{k-1} q_i(\mathcal{B}) s_i(H[I_Y](p)) \Big\|^2$$

CLIENT SIDE

RECIPE

INPUT IMAGE

OUTPUT IMAGE

Reconstruction of Output

- Recipe received from the server is applied on the high quality local input image so as to produce desired transformation.

Math behind Reconstruction

Output low-pass residual

$$L_n[O_c](p) = R_c(p)\big(L_n[I_c](p) + 1\big) - 1$$

Intensity levels of the multi Scale component

$$L_\ell[\hat{O}_Y] = m_\ell L_\ell[I_Y]$$

Chrominance

$$H_{\mathcal{B}}[O_{CC}](p) = \mathbf{A}_c(\mathcal{B})H[I](p) + \mathbf{b}_c(\mathcal{B})$$

Luminance

$$H_{\mathcal{B}}[O_Y](p) = \mathbf{A}_Y(\mathcal{B})H[I](p) + \mathbf{b}_Y(\mathcal{B}) + \sum_{i=1}^{k-1} q_i(\mathcal{B})s_i(H[I_Y](p))$$

Combine all the terms to assemble the reconstructed output

$$O = \mathrm{up}(L_n[O_{CC}]) + \hat{O}_Y + H[O_{CC}] + H[O_Y]$$

Original Results vs Our Results

Operator Applied	Results obtained from the paper	Our code
Laplacian operator	32.4 dB	29.78 dB
Time of the day*	37.2 dB	32.0 dB
Dehazing*	34.3 dB	35.17 dB
Warping	Failed	Failed

* represents that the images used for images are different due to unavailability of dataset

OUTPUTS(Laplacian Operator)

Input Image

Expected Output

Actual Output(29.78 dB)

OUTPUTS(Reverse Laplacian Operator)

Input Image

Expected Output

Actual Output (33.87 dB)

OUTPUTS(Time of the day)

Input Image

Expected Output

Actual Output(32.0 dB)

OUTPUTS(Dehazing)

Input Image

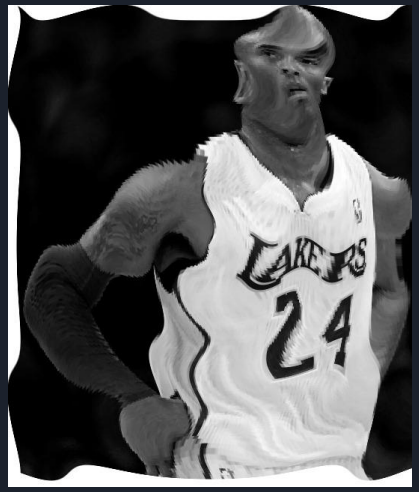
Expected Output

Actual Output(35.17 dB)

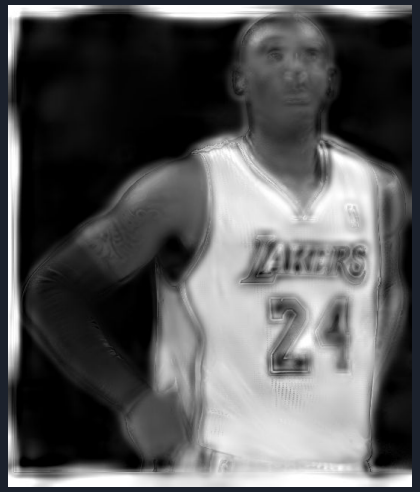
Restrictions on the Domain

OUTPUTS(Warping)

Input Image



Expected Output



Actual Output

Division of work

Abhishek - Recipe Construction and Upsampling

Kalyan - Reconstruction and Documentation

Shashank - Reconstruction and Image compression and Documentation

Jaiwanth - Image Compression and Upsampling and Documentation

Thank you! :)