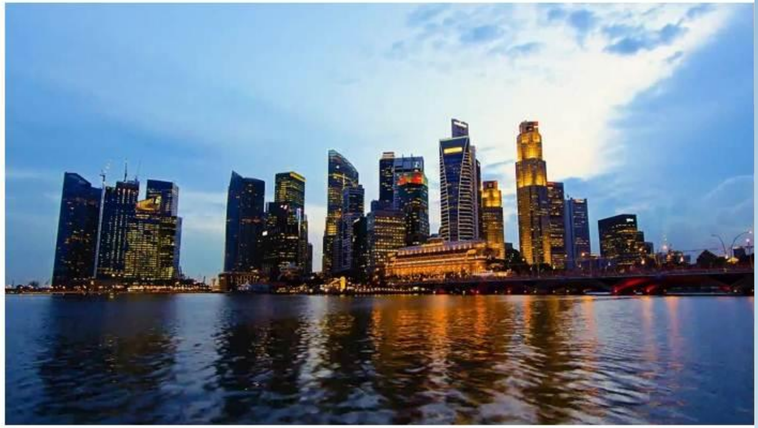
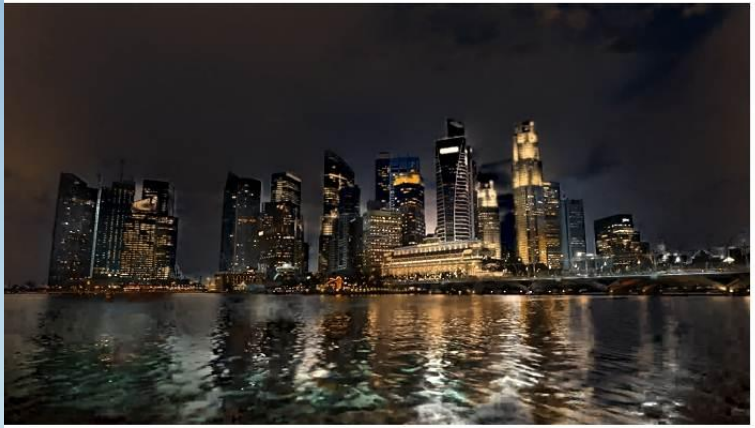
Data-driven Hallucination for Different Times of Day from a Single Outdoor Photo

Computer Vision

01
Problem Overview

Given **a single image as input**, we want to automatically create a plausible-looking photo that appears as though it was taken at a **different time of the day**. This should be done using a database of **time lapse videos**, of which the input image may not be a part of. Also we want to make the image look **realistic**, while **preserving the colour schema**.


Input image at "blue hour" (just after sunset)


Hallucinate at night

Most photographers cannot be at the **right place** at the **perfect time** and end up taking photos in the middle of the day when lighting is harsh. In this project, we implement an automatic technique that takes a single outdoor photo as input and seeks to **hallucinate an image of the same scene taken at a different time of day**.

02

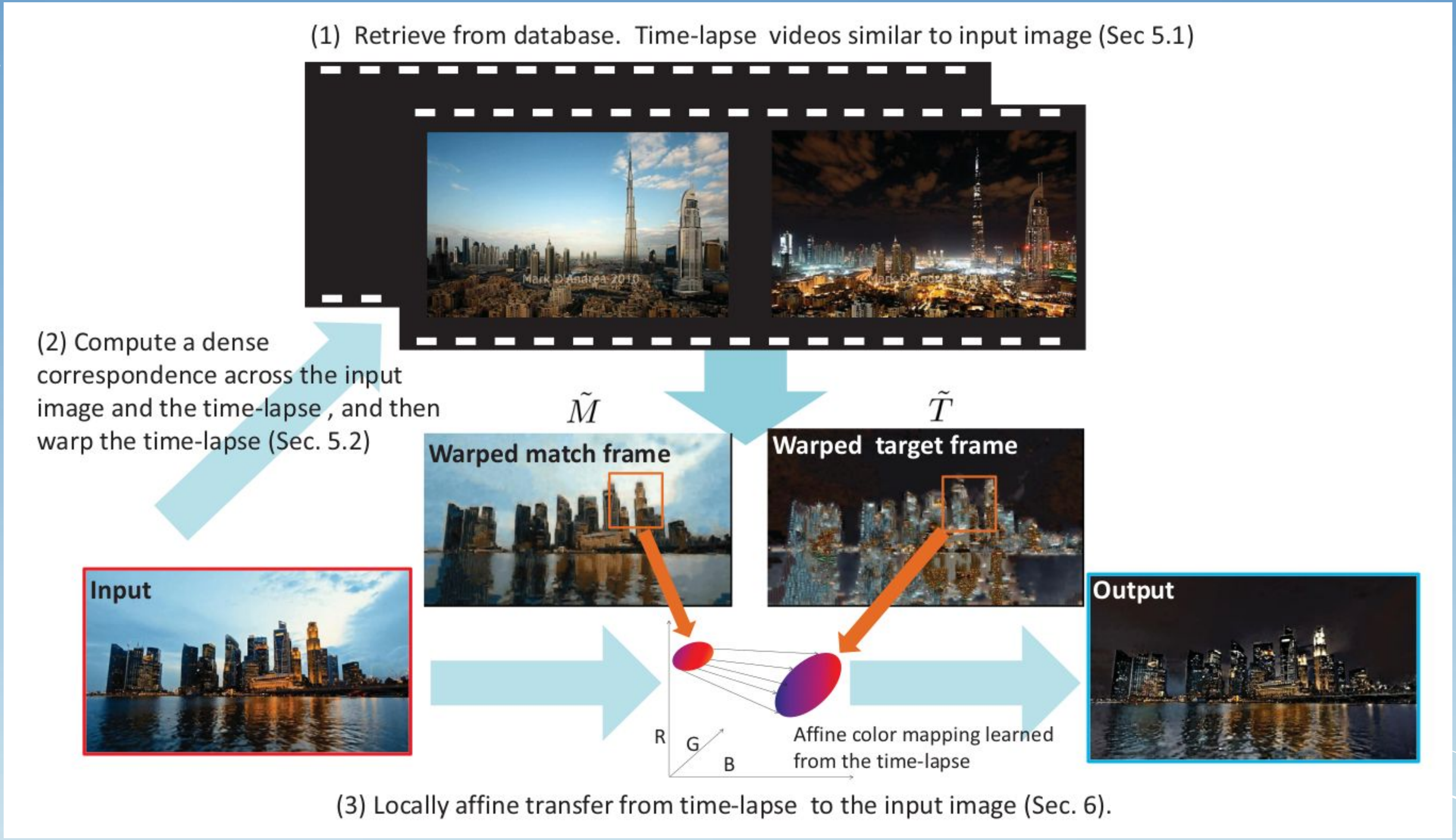
SOLUTION PROPOSED

Global Matching

Frame Selection

Local Matching

Local Affine Color Transfer


(1) Retrieve from database. Time-lapse videos similar to input image (Sec 5.1)

(2) Compute a dense correspondence across the input image and the time-lapse , and then warp the time-lapse (Sec. 5.2)

(3) Locally affine transfer from time-lapse to the input image (Sec. 6).

Global Matching

The first step of our algorithm is to identify the videos showing a scene similar to the given input image. A standard scene matching technique of computer vision is employed. We sample 5 regularly spaced frames from each video, and then compare the input to all these sampled frames. To assign a score to each time-lapse video, we use the **highest similarity score in feature space** of its sampled frames. We use **Histograms of Oriented Gradients (HOG) descriptor** to describe each frame.

Frame Selection

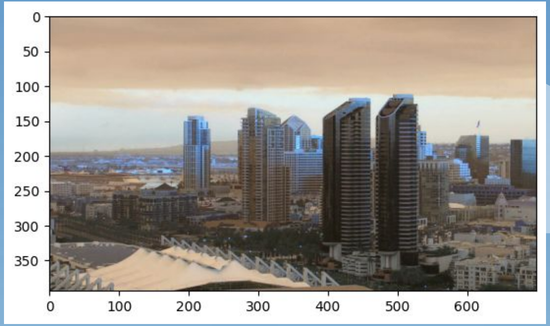
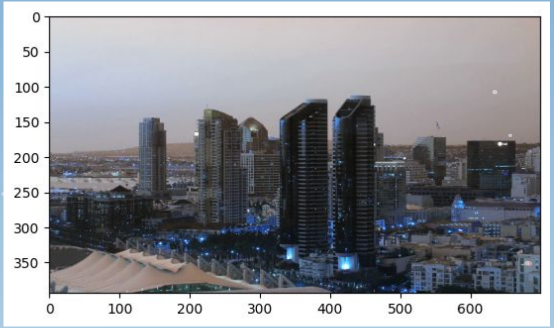
For each of the selected videos with a similar content as the input image, we seek to retrieve a frame that matches the time of day of the input image. We use the **color histogram** and **L2 norm** to pick the matched frame.

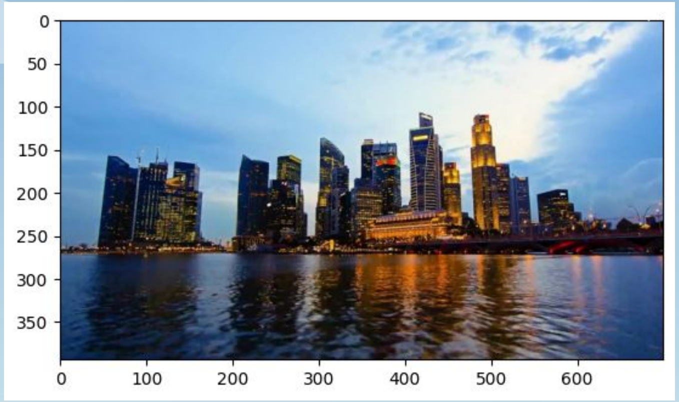
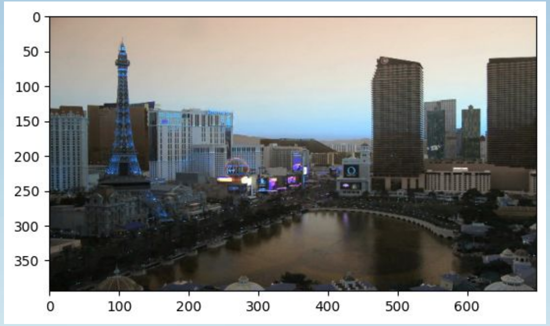
For each frame, we calculate the distance between the frame's histogram and the input image's histogram using the L2 norm. **The frame with the smallest histogram distance to the input image is selected as the best match**.

$$H(i) = [I_o, I_1, I_2, \ldots, I_{255}]$$

$$cost(x) = \sum_i (I_i - X_i)^2$$

$$M = X \ni (cost(X) = \min_{\forall y \in X} cost(y))$$

GRAND
planet hollywood
ph
Panasonic
BELLAGIO
O
CIRQUE DU SOLEIL



Local Matching

We seek to pair each pixel in the input image I with a pixel in the match frame M. To do this, we formulate the problem as a Markov Random Field (MRF) using a data term and pairwise term.

For the data term, we use the L2 norm over square patches of side length $2r + 1$.

For pairwise term, we leverage the information provided in the time-lapse video. We use L2 norm within each frame t, but L_∞ norm across frames so that the assigned compatibility score corresponds to the worst case over the video V .

Energy Function

Data Term

For each patch in I(Input images), we seek a patch in M(best matched frame) that looks similar to it. We use the L2 norm over square patches of side length 2r + 1. For pixel p ∈ I and the corresponding pixel q ∈ M , we calculate pixel wise difference in intensities for the patch centered around the pixel.

$$E_1 = \sum_{i=-r}^{+r} \sum_{j=-r}^{+r} \left\| I(x_p + i, y_p + j) - M(x_q + i, y_q + j) \right\|^2$$

Pairwise Term

They leverage the information provided in a time-lapse video. For two adjacent pixels pi and pj in I, we name Ω the set of the overlapping pixels between the two patches centered at pi and pj . For each pixel o ∈ Ω, we define the offsets δi = o − pi and δj = o − pj. For the energy we use L2 norm within each frame t, but L∞ norm across frames so that the assigned compatibility score corresponds to the worst case over the video V .

$$E_2(q_i, q_j) = \max_t \sum_{o \in \Omega} \left\| V_t(q_i + \delta_i) - V_t(q_j + \delta_j) \right\|^2$$

Local Affine Transfer

The transfer from the input image I, the warped match frame M, the warped target frame T, and output the hallucinated image O involves computing affine model Ak for each of the k patches.

A naive solution would be to compute each affine model Ak as a regression between the kth patch of M and its counterpart in T, and then independently apply Ak to the kth patch of I for each k. However, the boundary between any two patches of O would not be locally affine with respect to I, and would make O have a different structure from I, e.g., allows for spurious discontinuities to appear at patch boundaries.

Instead of this naive approach, this problem can be formulated as a least-squares optimization that seeks local affinity everywhere between O and I.

$$\sum_k \left\| \mathbf{v}_k(\tilde{T}) - \mathbf{A}_k \, \bar{\mathbf{v}}_k(\tilde{M}) \right\|_{\mathsf{F}}^2$$

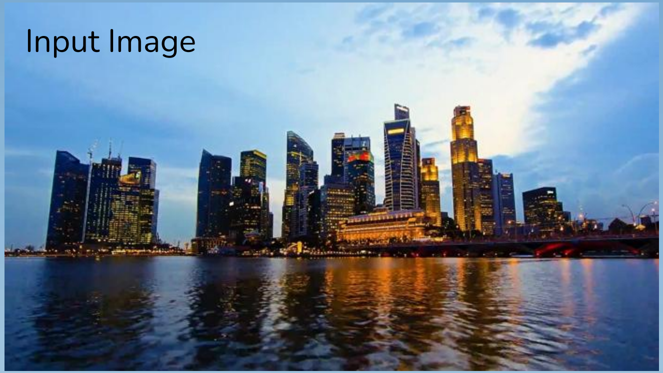
$$\sum_k \left\| \mathbf{v}_k(O) - \mathbf{A}_k \, \bar{\mathbf{v}}_k(I) \right\|_{\mathsf{F}}^2$$

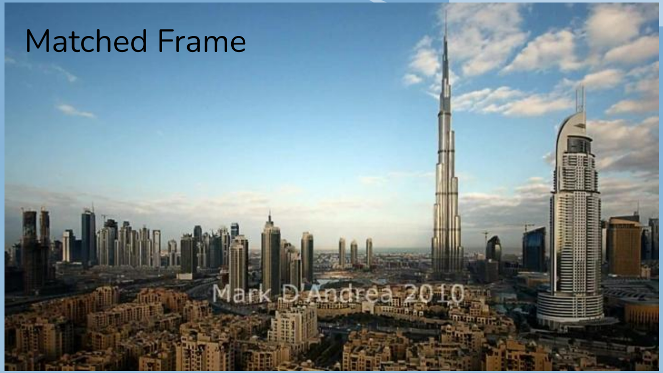
$$\begin{aligned} O = \arg\min_{O,\{\mathbf{A}_k\}} & \sum_k \left\| \mathbf{v}_k(O) - \mathbf{A}_k \, \bar{\mathbf{v}}_k(I) \right\|^2 \\ & + \epsilon \sum_k \left\| \mathbf{v}_k(\tilde{T}) - \mathbf{A}_k \, \bar{\mathbf{v}}_k(\tilde{M}) \right\|^2 + \gamma \sum_k \left\| \mathbf{A}_k - \mathbf{G} \right\|_{\mathsf{F}}^2 \end{aligned}$$

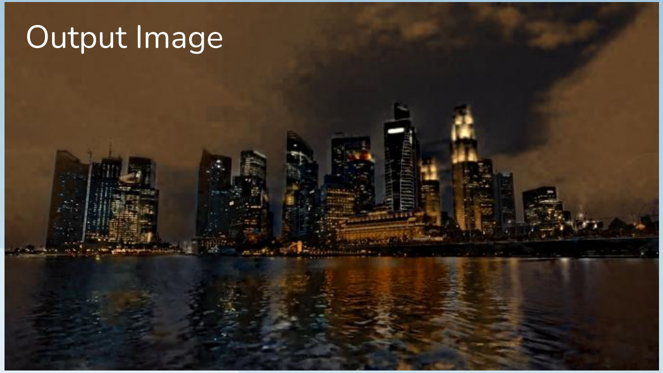
03

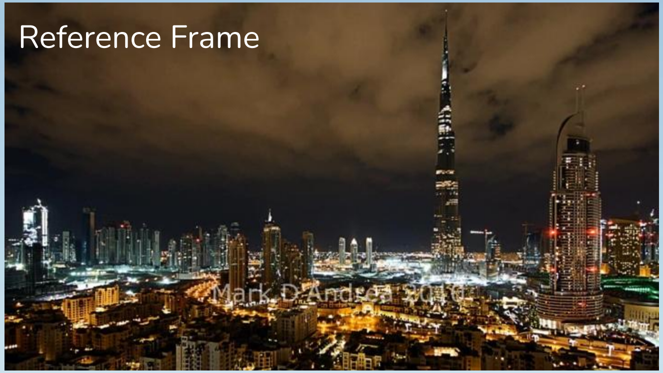
OUR IMPLEMENTATION

Results

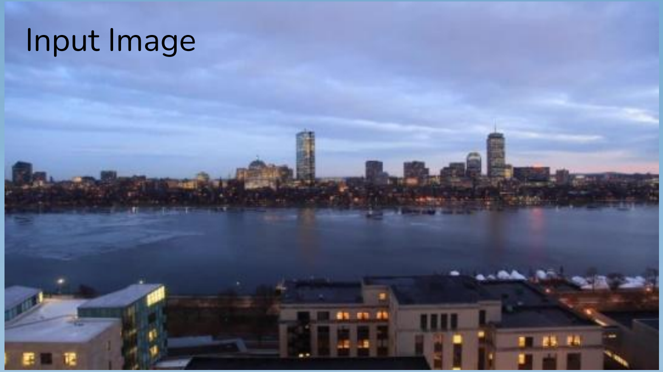

Input Image

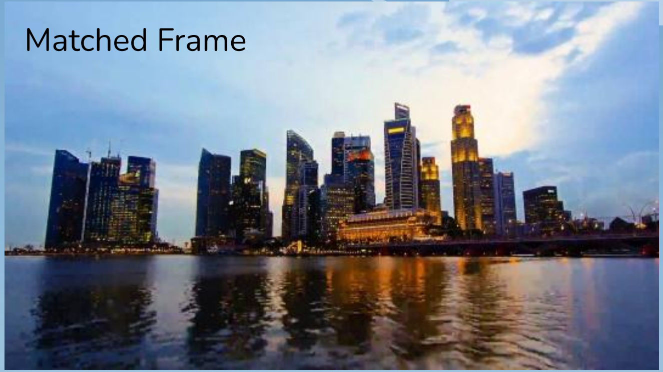

Matched Frame
Mark D'Andrea 2010

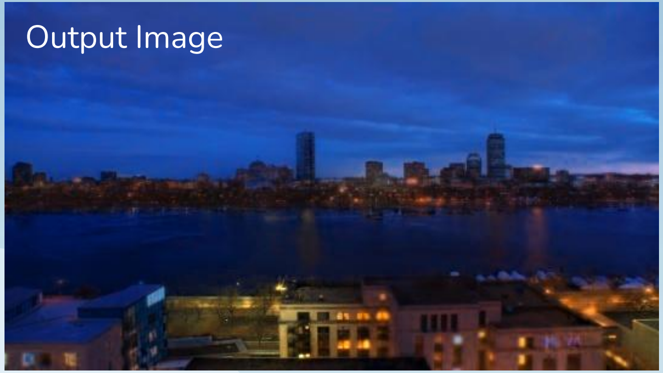

Output Image

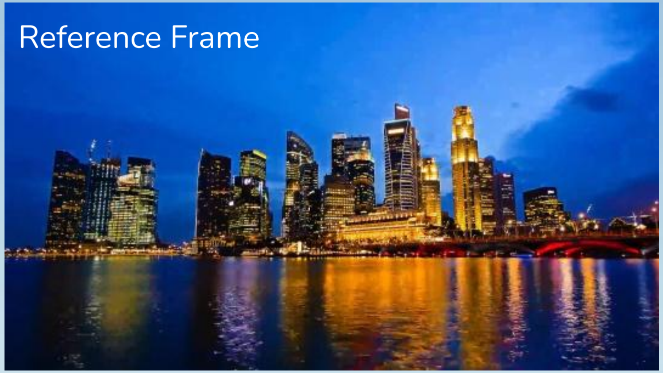

Reference Frame
Mark D'Andrea 2010

Results

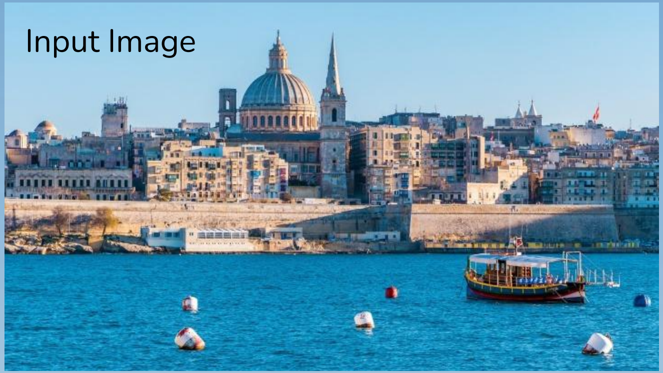


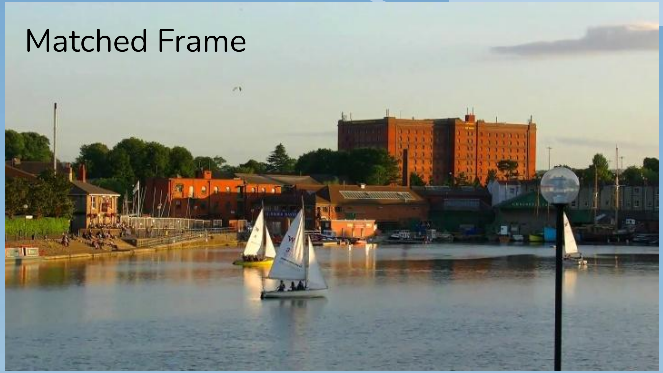


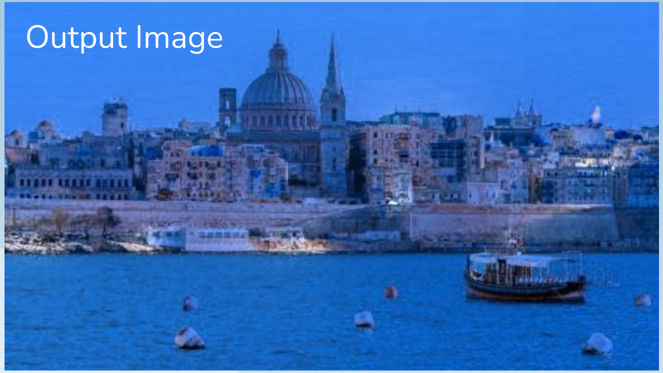


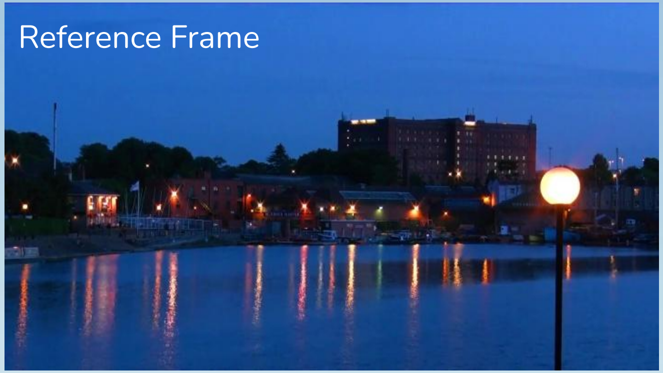


Results



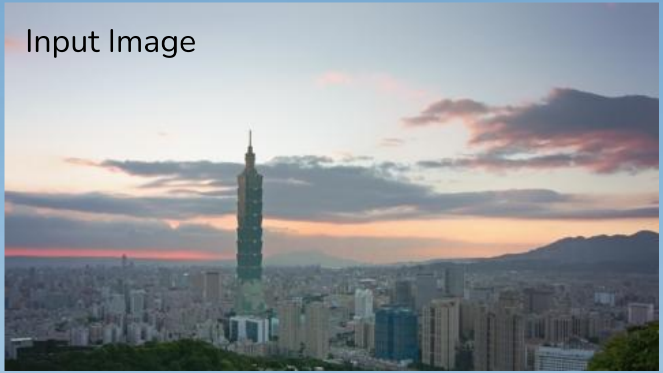




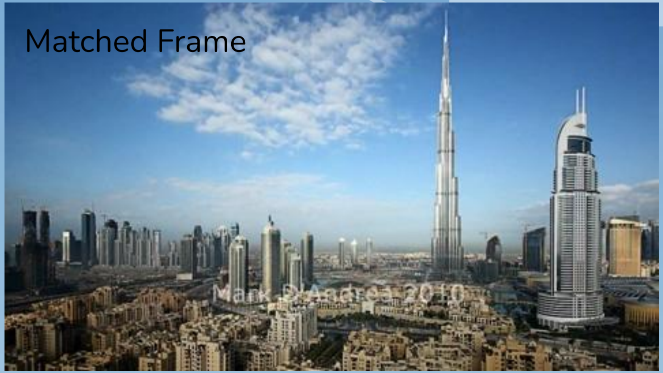


Results

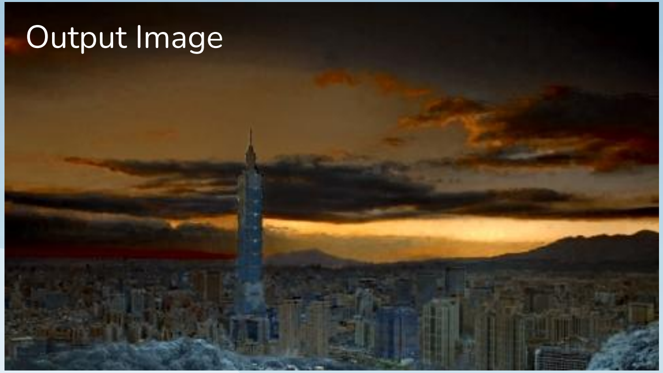
Input Image



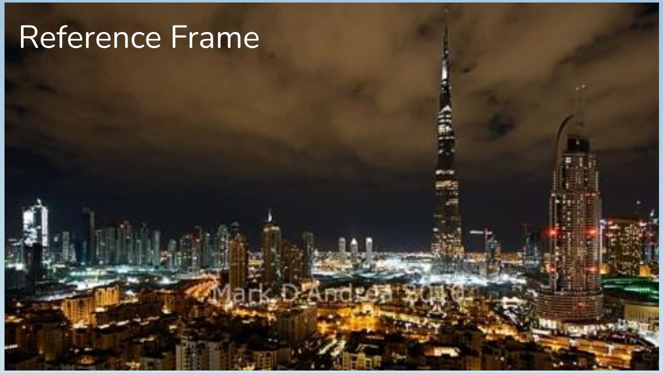
Matched Frame



Output Image



Reference Frame

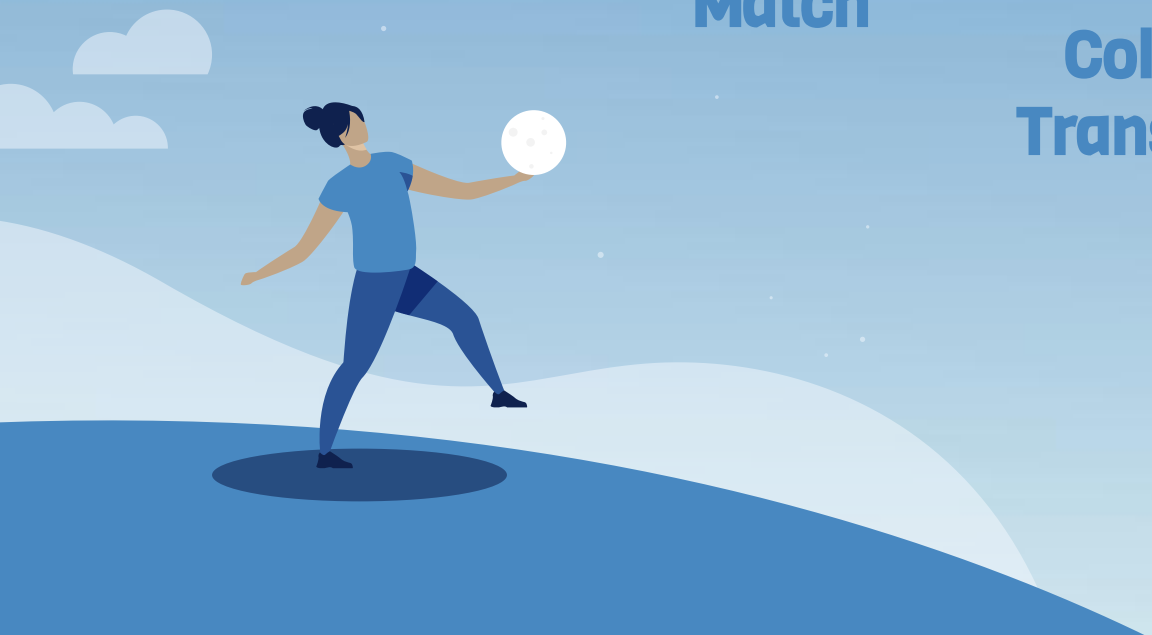


Novelty/Experiments

Patch Match

Color Transfer

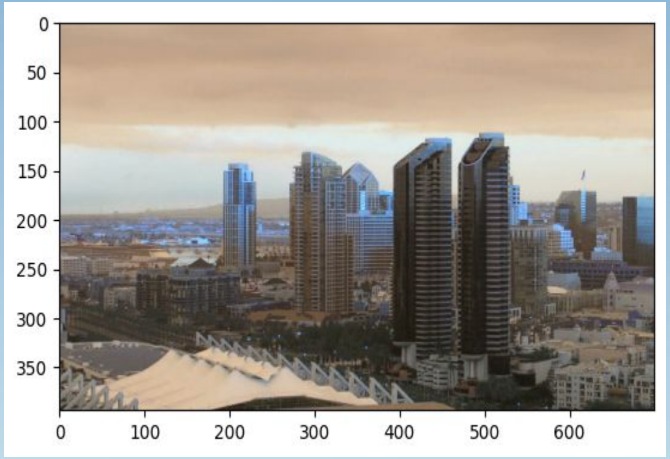
SIFT Descriptor



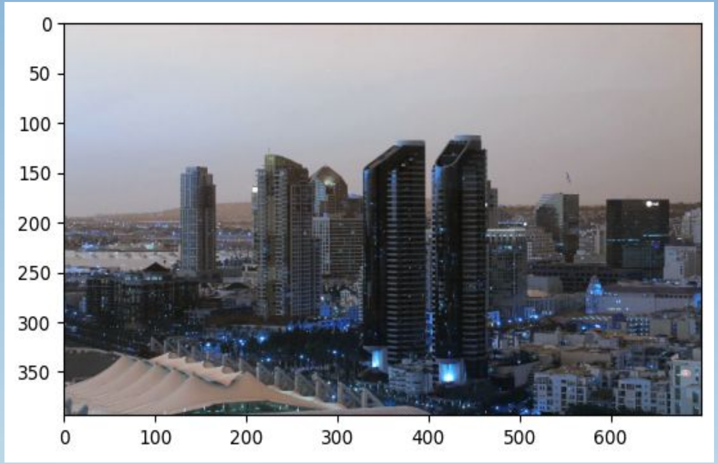
SIFT Descriptor

We experimented with SIFT descriptor(in place of HOG) for calculating the similarity between the input image and the sampled frames.

Matching frame using HOG



Matching frame using SIFT



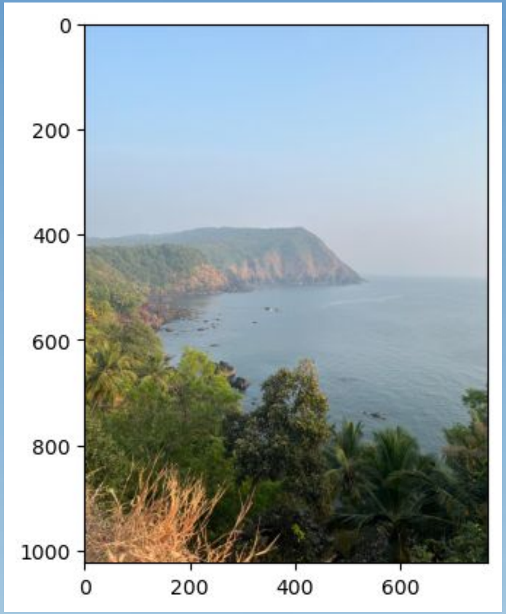
Different videos showing a scene similar to the given input image

PatchMatch

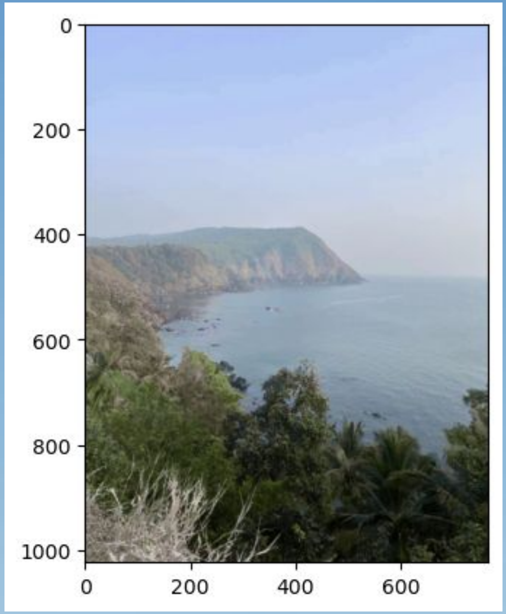
- PatchMatch is an algorithm for fast nearest neighbor field computation for image patches. It works by finding, for each pixel in an input image, the most similar pixel in a second image, using an iterative approach.
- The algorithm takes as input two images, x and y, and outputs a field f, which describes for each pixel in x the corresponding pixel in y that has the most similar patch.
- The PatchMatch algorithm consists of three main steps: initialization, propagation, and random search.
- 1. In the **initialization step**, each pixel in the input image is assigned a random corresponding pixel in the second image.

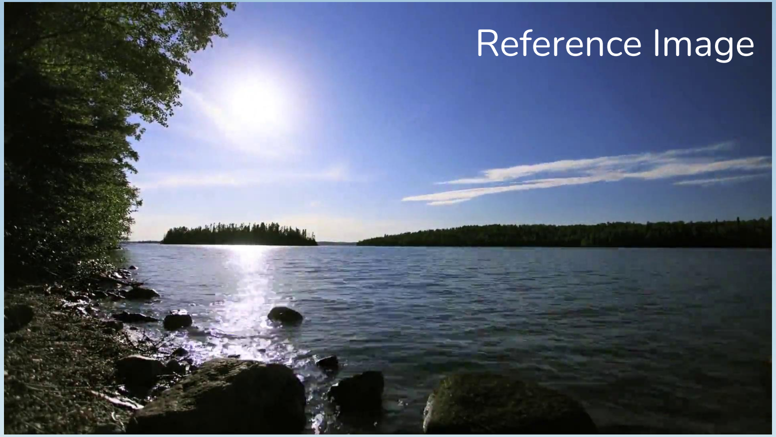
  2. In the **propagation step**, the algorithm iteratively updates the corresponding pixel for each input pixel, using information from neighboring pixels. We find a better corresponding pixel by comparing the patch surrounding the current pixel in the input image to patches surrounding candidate pixels in the second image.

  3. In the random search step, the algorithm performs a local search around each pixel to try and find a better corresponding pixel.

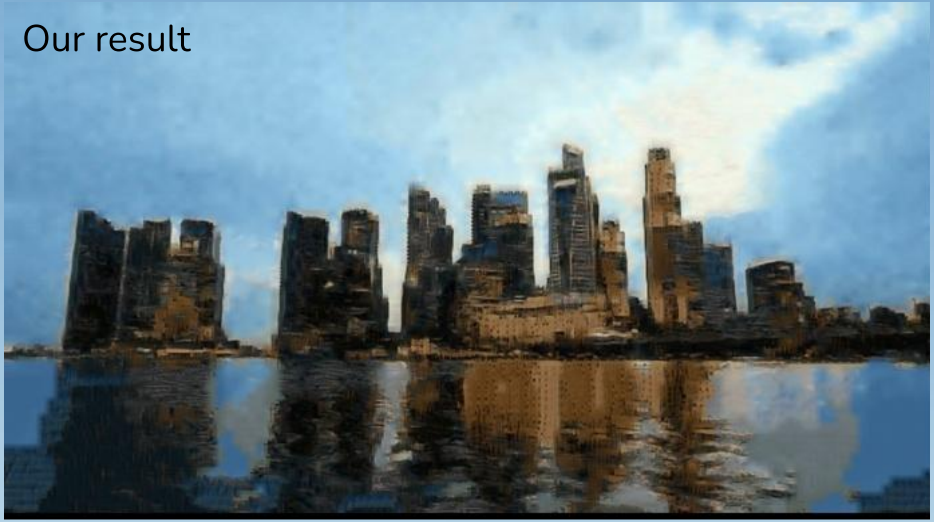


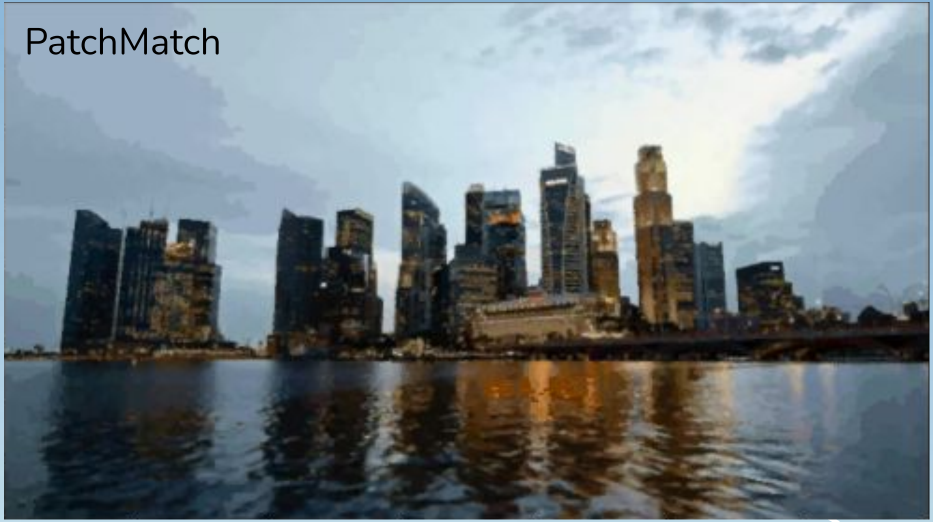
Reconstruction





Reconstruction of Input image using the Best Matched Frame

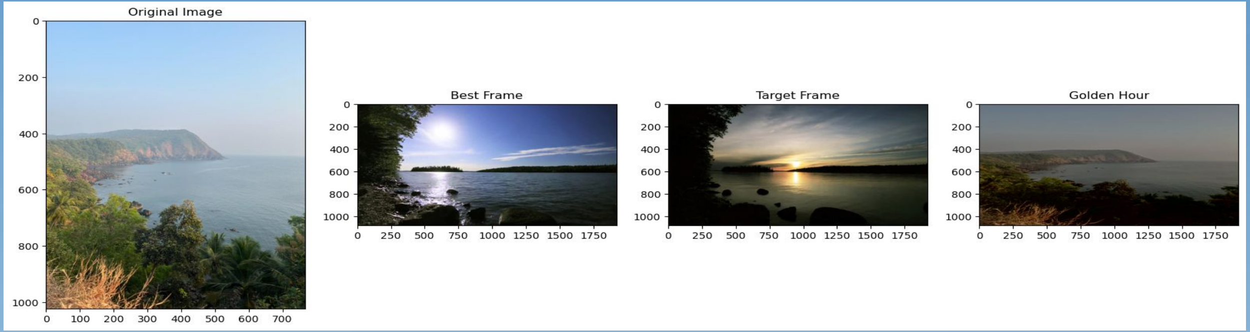


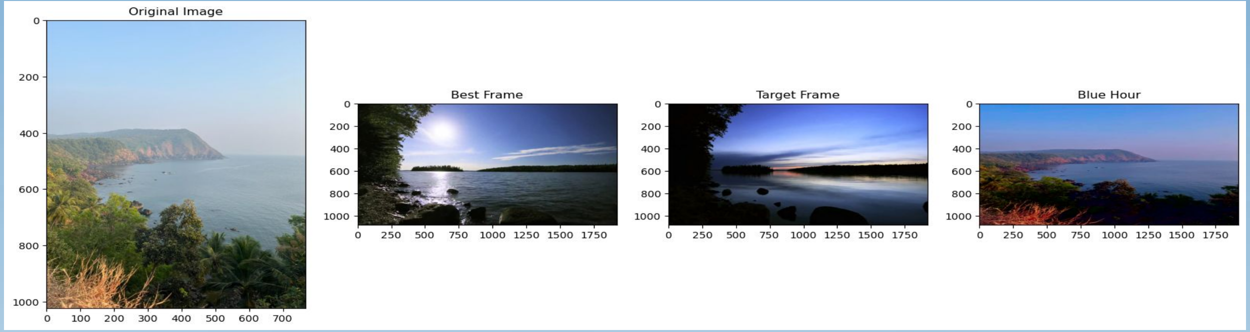


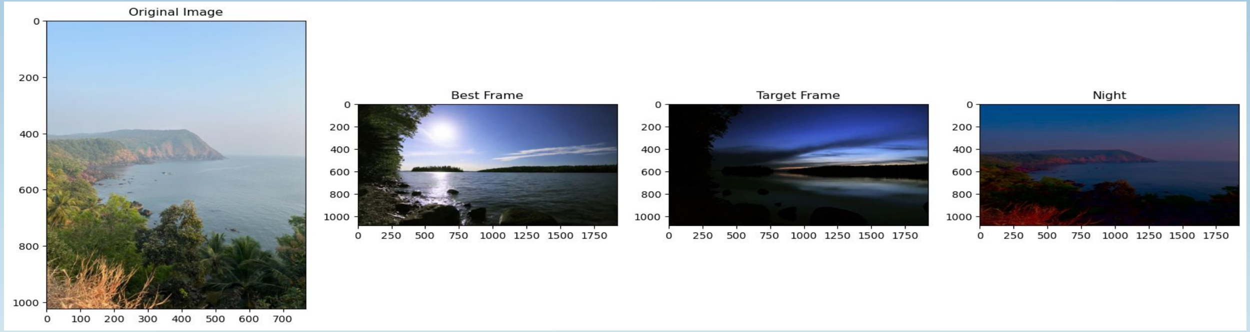
Color Transfer

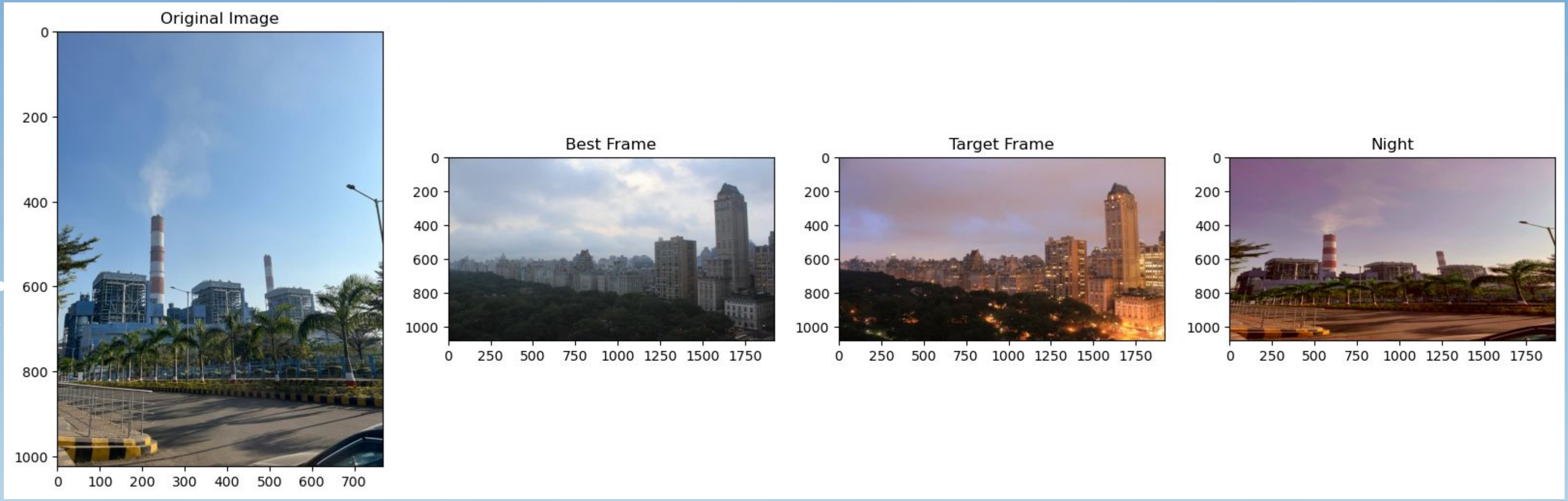
- Color transfer is a practical method to change the appearance of a source image **according to the color pattern of a target image**. This methods uses the **LAB color space**.
- The input and target images are converted from the default BGR color space to the LAB color space.
- The function then **calculates the mean and standard deviation of each color channel** in the input and target images.
- Finally, the function performs the color transfer by looping through each pixel in the input image, **calculating the new color value for each channel using the color statistics of the target image**, and rounding and clipping the result to ensure that the new value is within the range of 0-255.

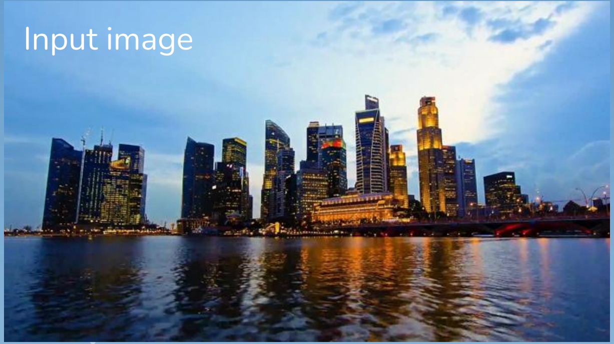
$$I_k = \frac{\sigma_t^k}{\sigma_s^k}(S^k - mean(S^k)) + mean(T^k) \ , k = (\ell, \alpha, \beta)$$

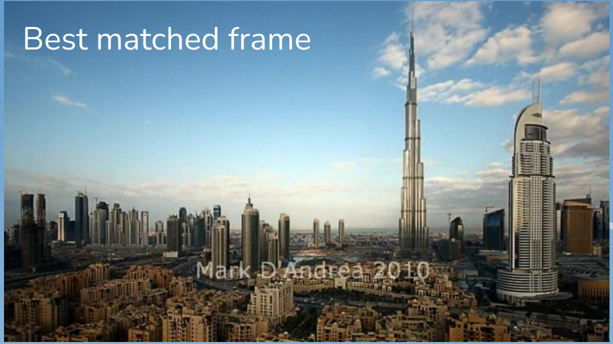

Original Image
Best Frame
Target Frame
Golden Hour

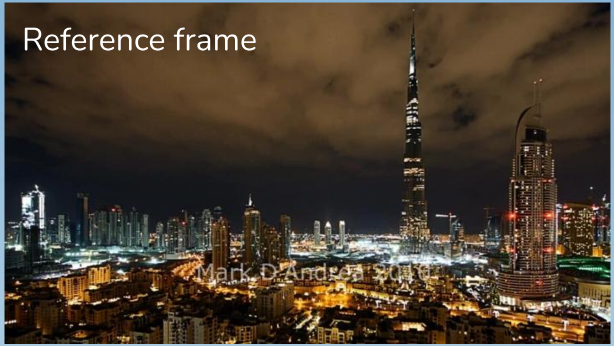

Original Image
Best Frame
Target Frame
Blue Hour

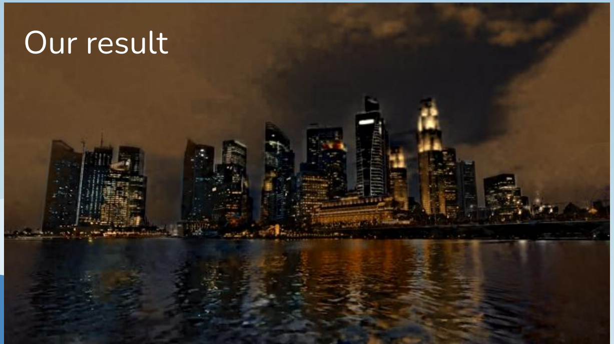

Original Image
Best Frame
Target Frame
Night

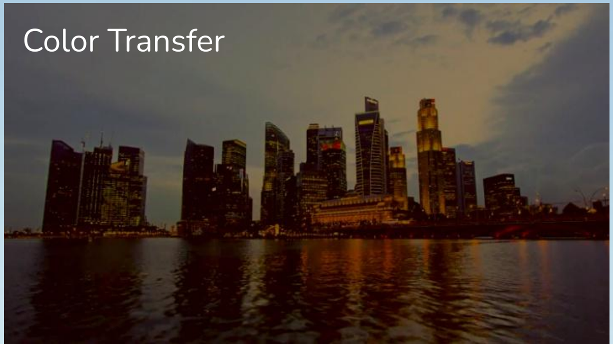

Original Image
Best Frame
Target Frame
Night
0
200
400
600
800
1000
0
100
200
300
400
500
600
700
0
250
500
750
1000
1250
1500
1750


Input image


Best matched frame
Mark D'Andrea 2010


Reference frame
Mark D'Andrea 2010


Our result


Color Transfer

THANKS!

Team SeaWeed

Divya Varshini
Souvik Karfa
Soveet Nayak