Image Edit

- Dehaze（图像去雾）
- SeamCarving（图像缩放）
- ReColor（图像上色）
- DeColor（图像去色）

Dehaze

- 有雾的图像



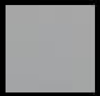



图像去雾的目标

- **Scene Restoration & Depth Estimation**

depth

Haze Imaging Model



Atmospheric light

$$\mathbf{I} = \mathbf{J} \cdot t + \mathbf{A} \cdot (1 - t)$$

Hazy image

Scene radiance

Transmission

Haze Imaging Model

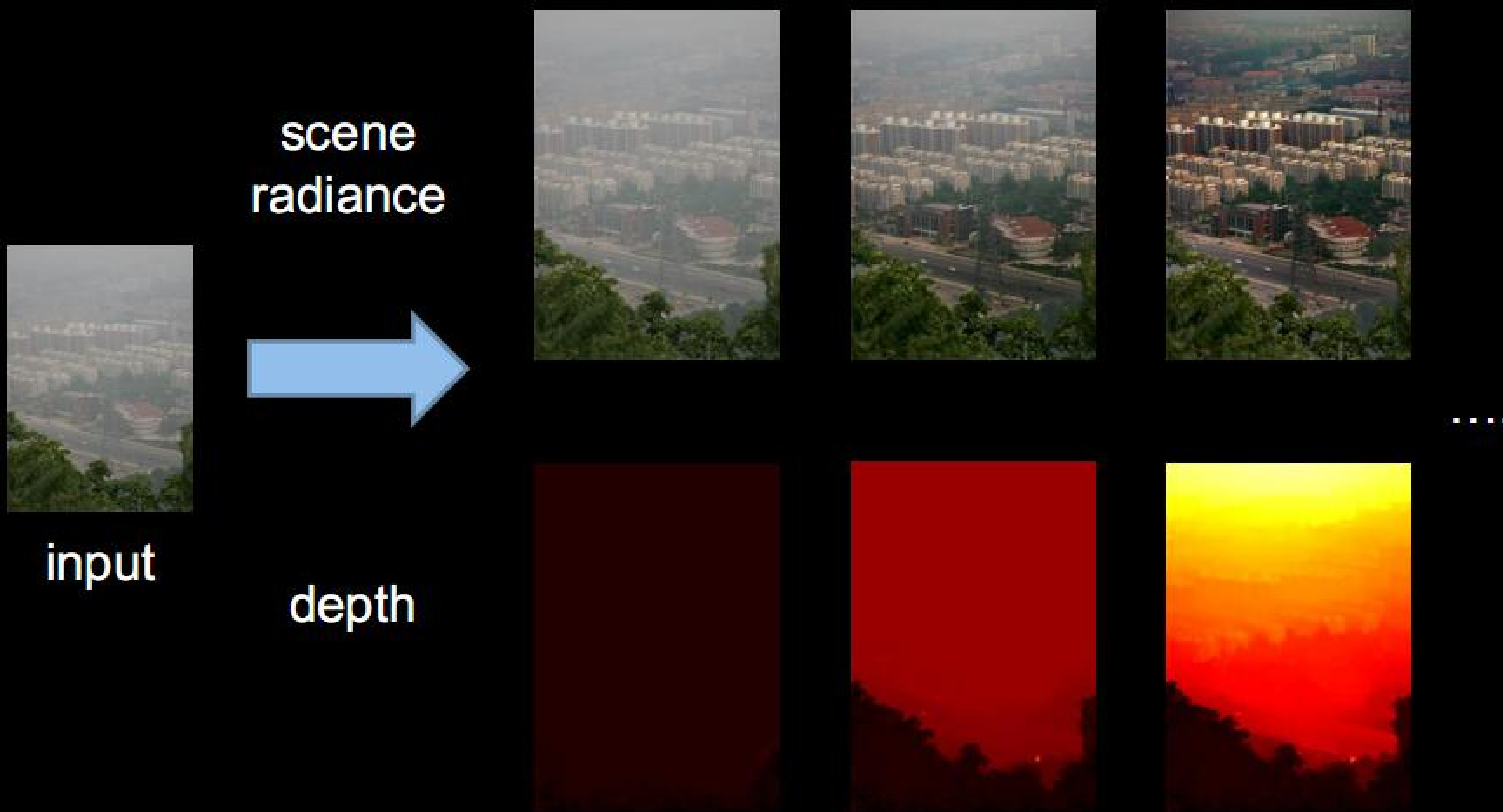
$$\mathbf{I} = \mathbf{J} \cdot t + \mathbf{A} \cdot (1 - t)$$

Depth

$$d = -\beta \ln t$$

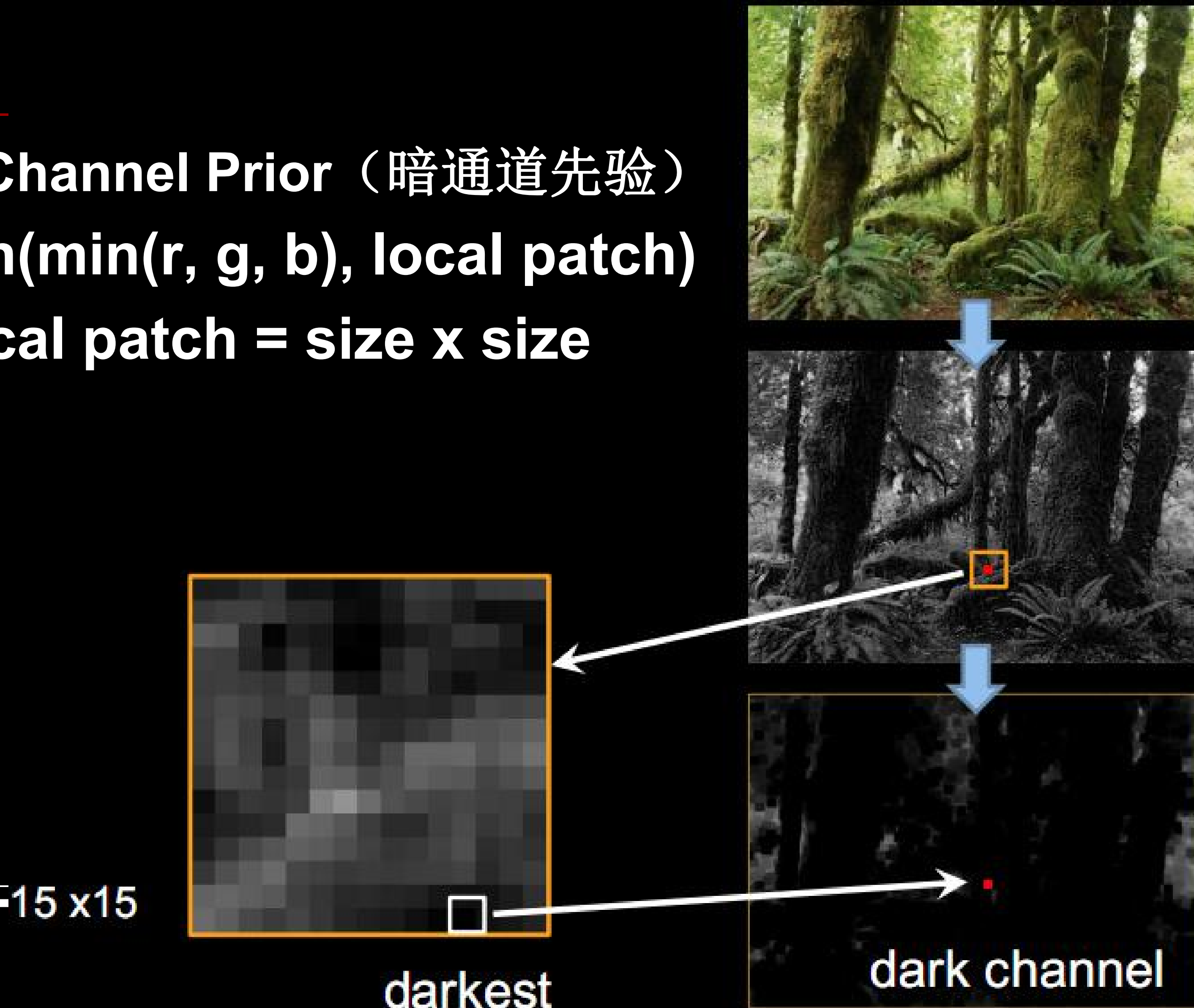
Transmission

去雾计算的病态问题



Priors 增加先验知识

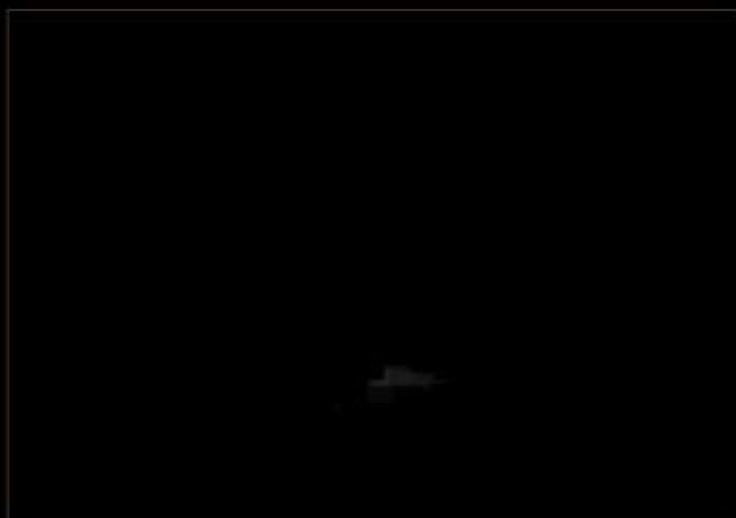
- Dark Channel Prior（暗通道先验）
  - min(min(r, g, b), local patch)
  - Local patch = size x size

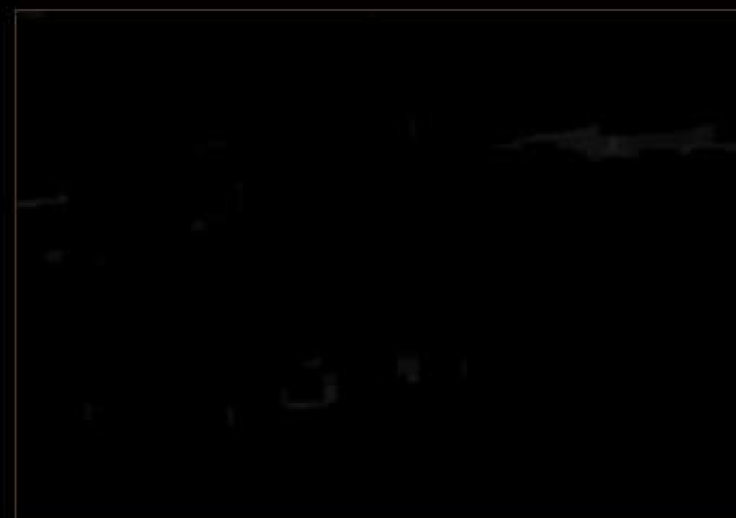


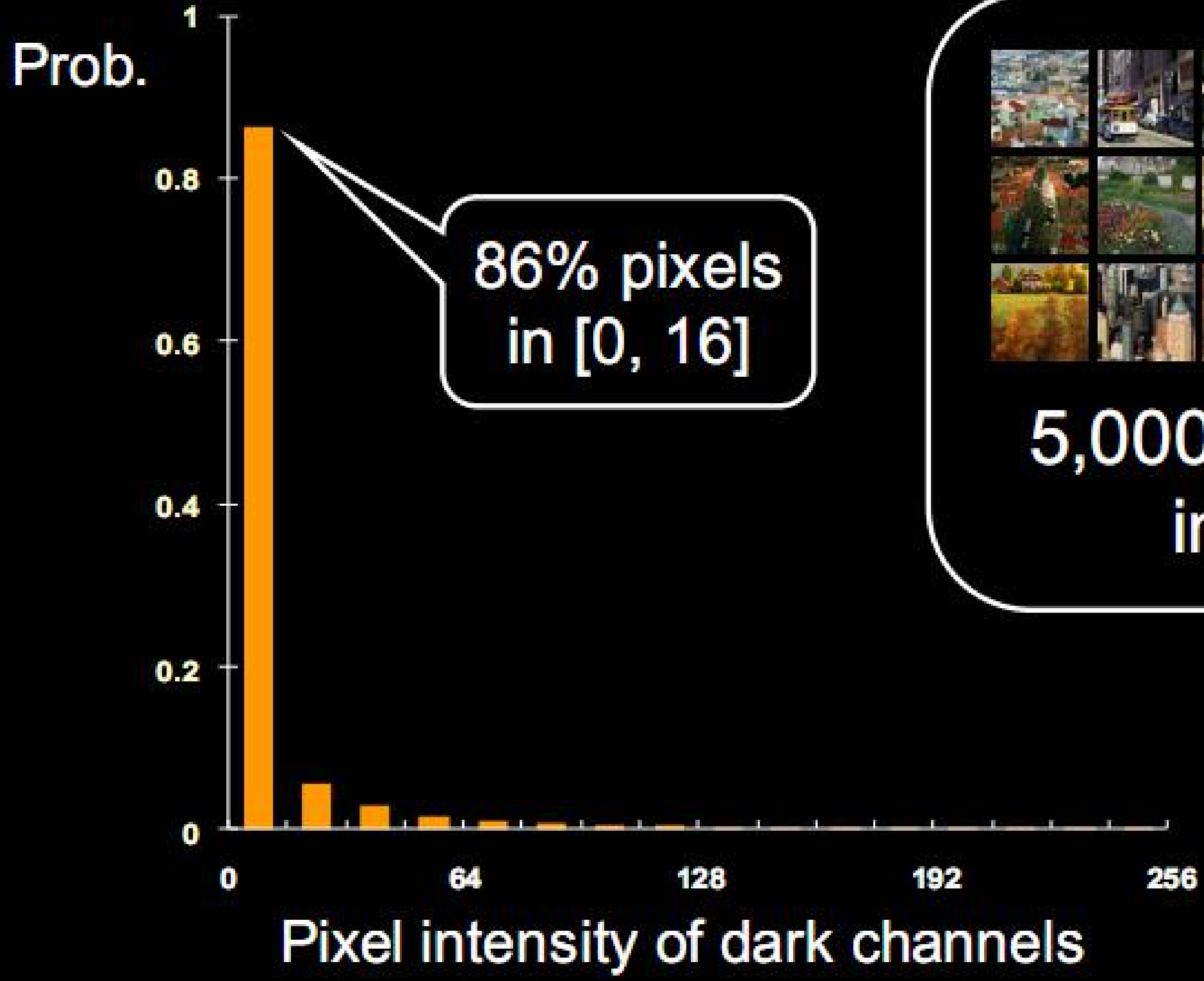
有趣的现象

Dark Channel


Dark Channel







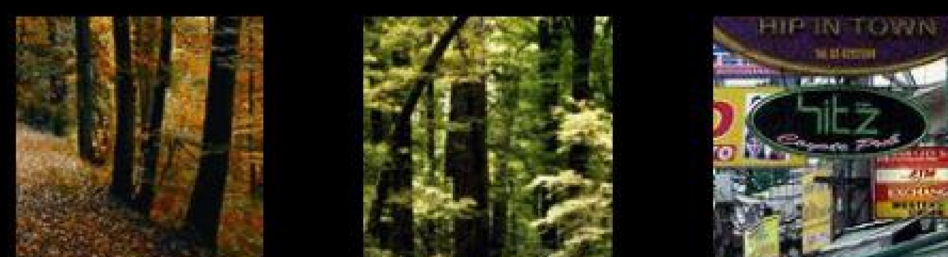
5,000 haze-free images

What makes it dark?

- Shadow

- Colorful object

- Black object



Dark Channel of Hazy Image

hazy image

dark channel

Transmission Estimation

Haze imaging model $\mathbf{I} = \mathbf{J} \cdot t + \mathbf{A} \cdot (1 - t)$

Normalize $\dfrac{\mathrm{I}^c}{\mathrm{A}^c} = \dfrac{\mathrm{J}^c}{\mathrm{A}^c} t + 1 - t$

Compute dark channel

$$\min_{\Omega} (\min_{c} \frac{\mathrm{I}^c}{\mathrm{A}^c}) = \left\{ \min_{\Omega} (\min_{c} \frac{\mathrm{J}^c}{\mathrm{A}^c}) \right\} t + 1 - t$$

Transmission Estimation

Dark Channel Prior

$$\min_{\Omega}\left(\min_{c} \mathrm{J}^{c}\right) \rightarrow 0$$

Compute dark channel

$$\min_{\Omega}\left(\min_{c} \frac{\mathrm{I}^{c}}{\mathrm{A}^{c}}\right) = \overset{\rightarrow 0}{\boxed{\left\{\min_{\Omega}\left(\min_{c} \frac{\mathrm{J}^{c}}{\mathrm{A}^{c}}\right)\right\}}} t + 1 - t$$

Transmission Estimation

Estimate transmission

$$t = 1 - \min_{\Omega}\left(\min_{c} \frac{\mathrm{I}^c}{\mathrm{A}^c}\right)$$

Compute dark channel

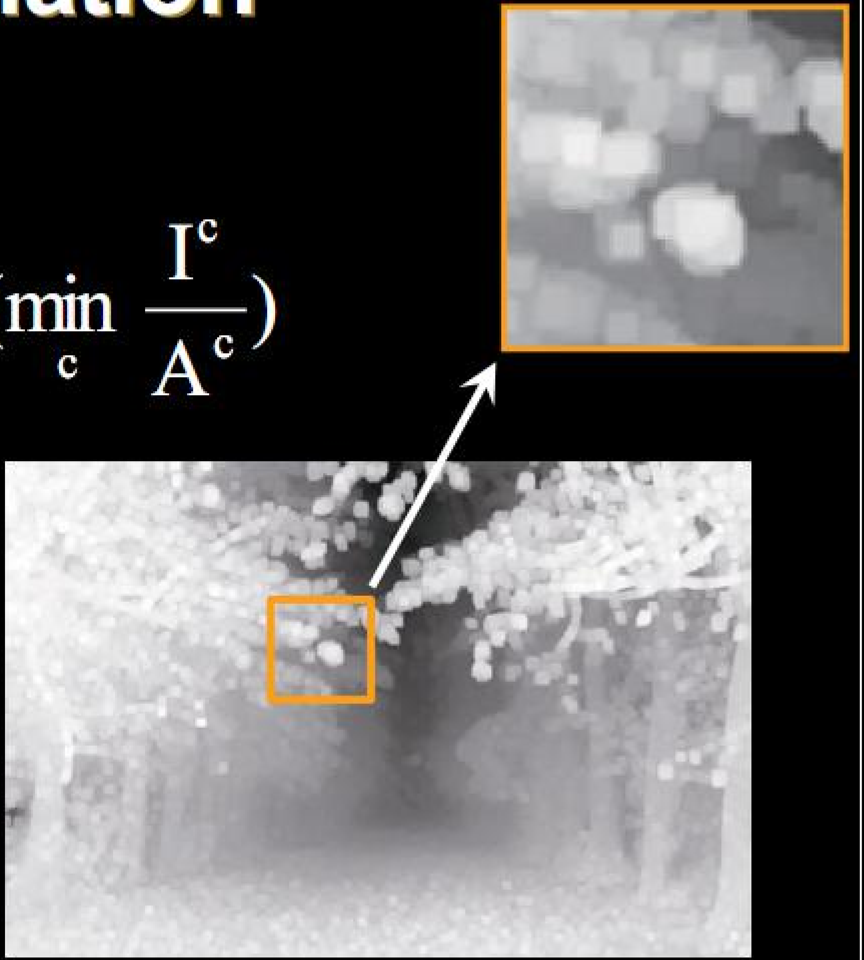
$$\min_{\Omega}\left(\min_{c} \frac{\mathrm{I}^c}{\mathrm{A}^c}\right) = \left\{\min_{\Omega}\left(\min_{c} \frac{\mathrm{J}^c}{\mathrm{A}^c}\right)\right\} t + 1 - t$$

Transmission Estimation

Estimate transmission

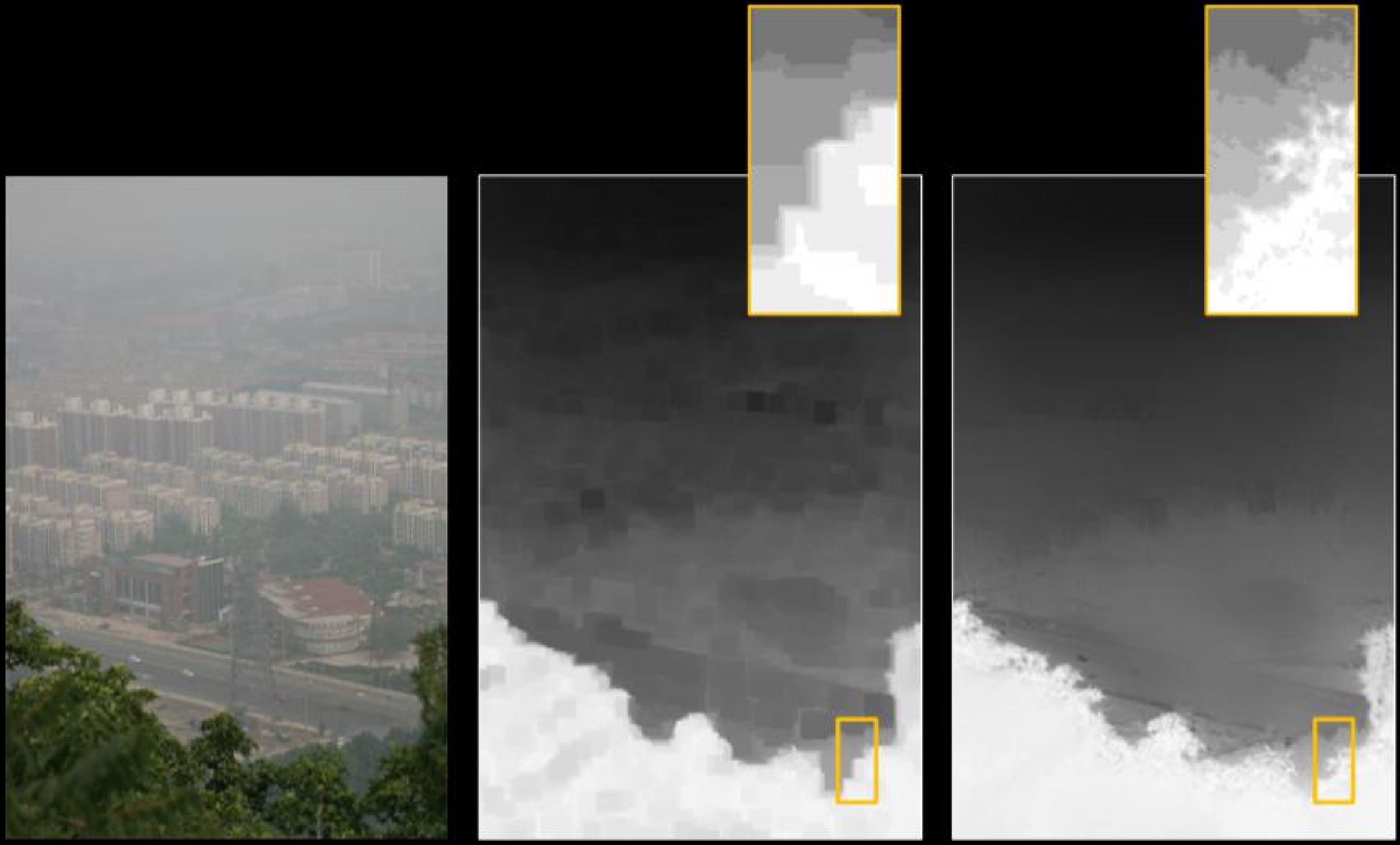
$$t = 1 - \min_{\Omega} \left(\min_{c} \frac{I^c}{A^c} \right)$$

input I

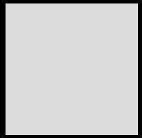


estimated t

优化投射图t（导向图滤波）



Scene Radiance Restoration



Atmospheric
light

$$\mathbf{I} = \mathbf{J} \cdot t + \mathbf{A} \cdot (1 - t)$$

Hazy image

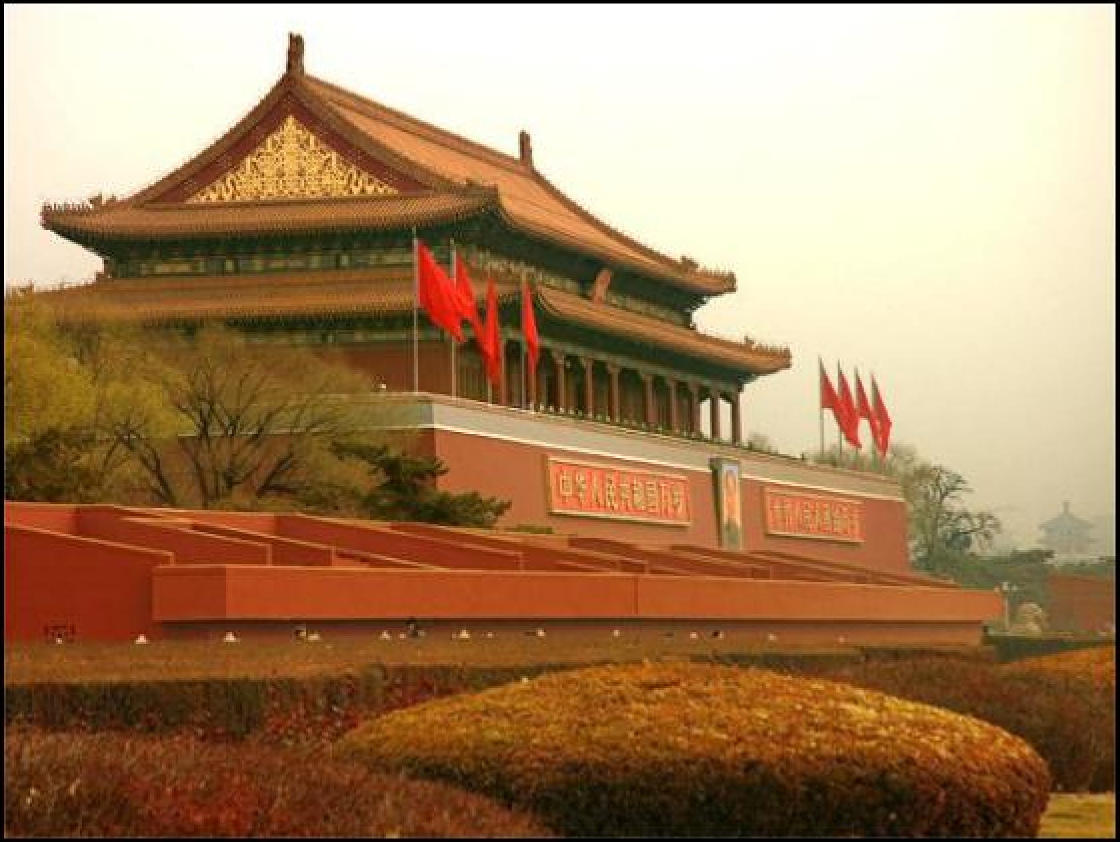
Scene radiance

Transmission

Results

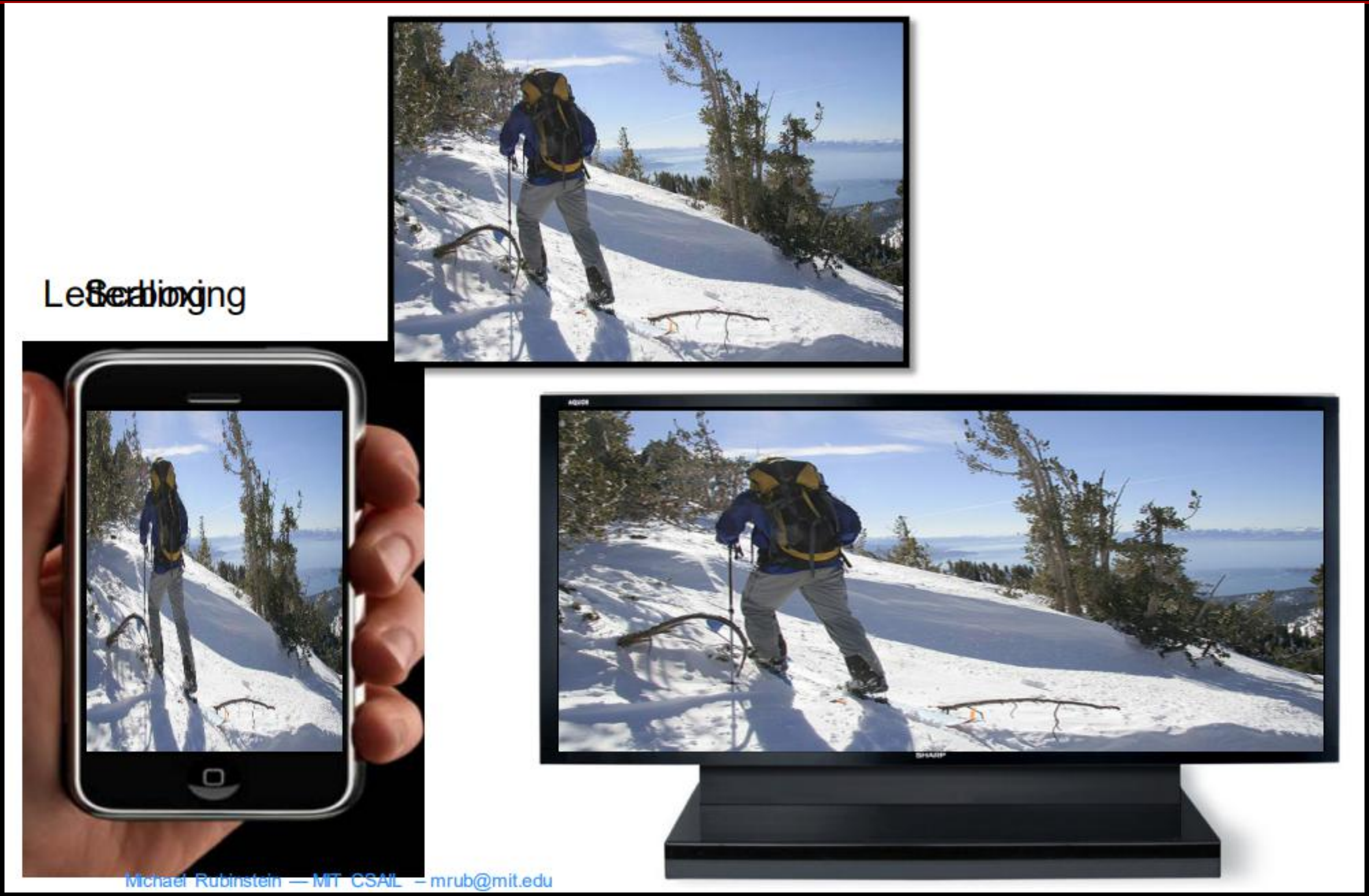
input

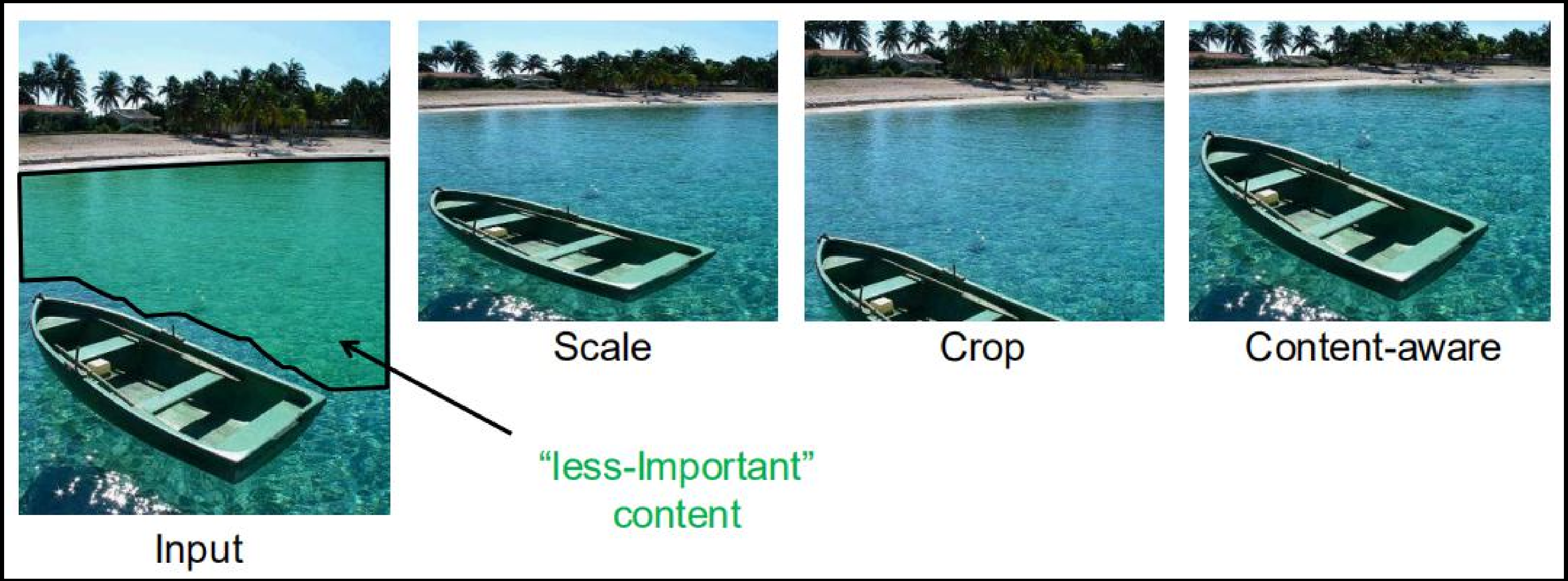
Results



recovered image

SeamCarving




Scale
Crop
Content-aware
"less-Important"
content
Input

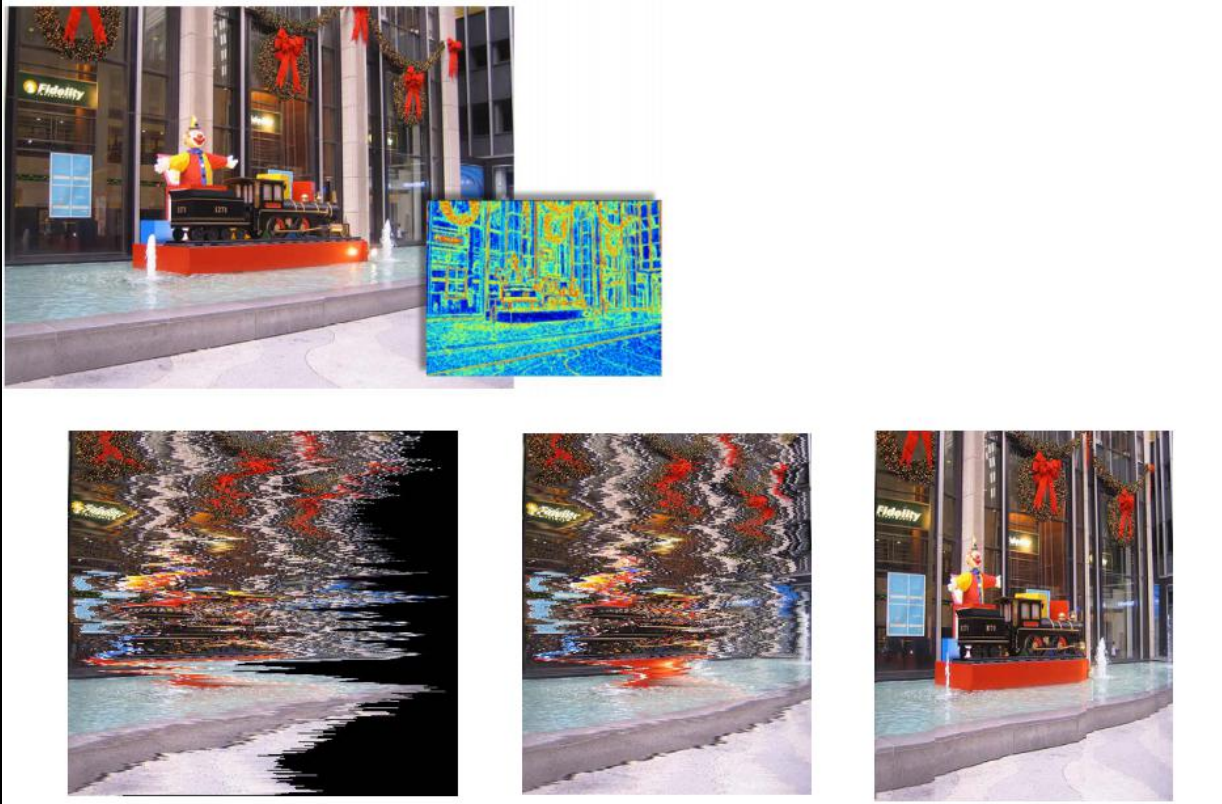
Goal

- 修改图像的比例
- 保持主体内容和结构尽量不变
- 尽量避免视觉上明显的瑕疵

- 输入一张尺寸 M x N 图像
- 输出一张 M x N' 图像，预先假定 N' < N

- 方法：从图像删除不重要的像素
  - 衡量重要性？
    - Sobel 计算

怎么删，删哪些？



Optimal

Least-energy pixels (per row)

Least-energy columns

寻找一条路径Seam

- 垂直 Seam
  - 从图形的顶部到底部的一条路径（8-联通），每一行存在且只存在一个像素

其他能量表示方法

- Saliency（显著性）

怎么快速计算？

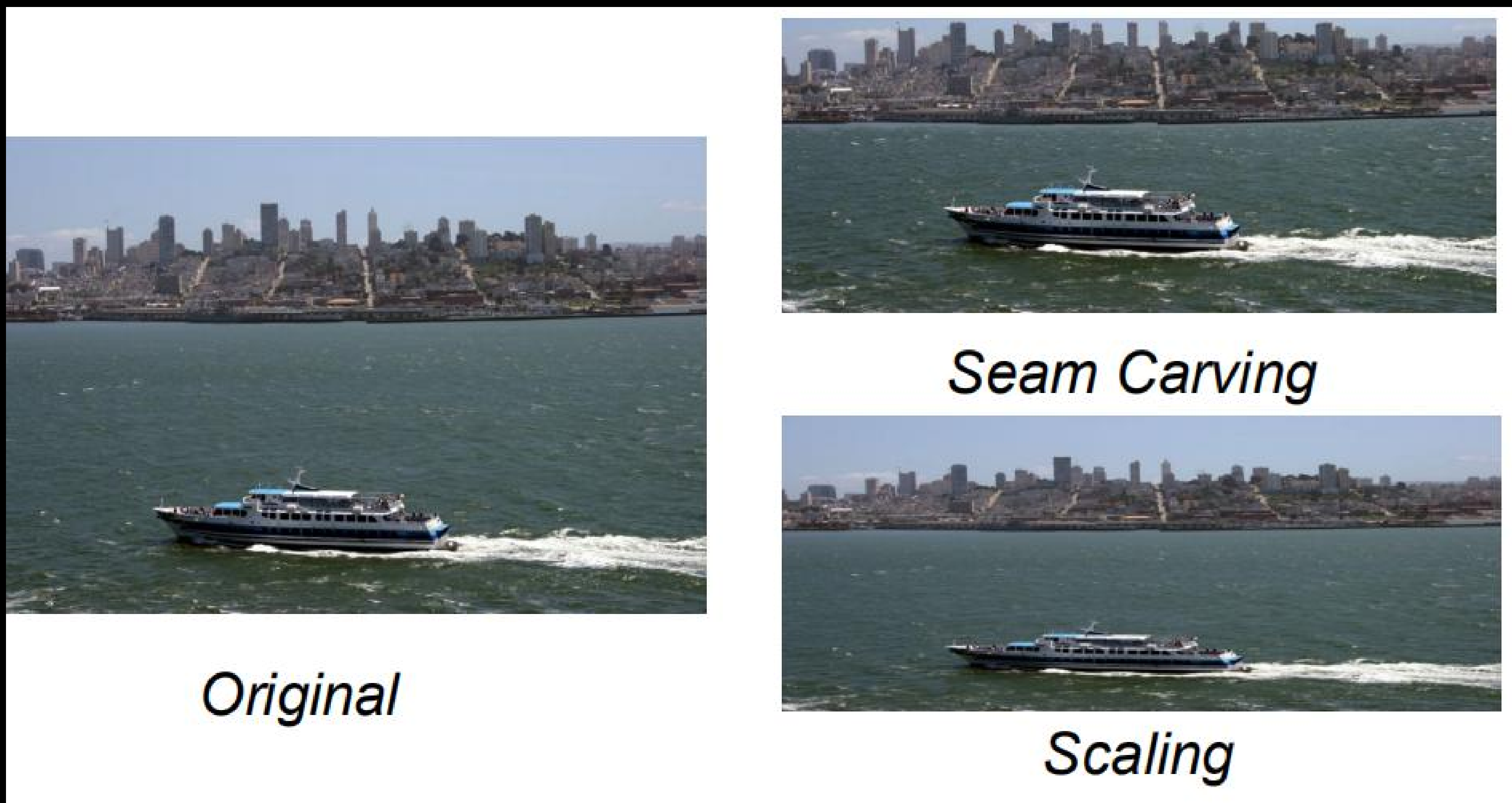
- 动态规划

5	8	12	3
9	2	3	9
7	3	4	2
4	5	7	8

5	8	12	3
9	7	6	12
14	9	10	8
14	14	15	16

5	8	12	3
9	7	6	12
14	9	10	8
14	14	15	16

结果



一些效果，How?

- 物体变大

- 图像加宽

- 物体移除

进一步思考

- 如何加快实时处理速度？

Colorization

Gray Images

- YUV Color Space, but U, V are zero

The Approach

- Two neighboring pixels **r**, **s** should have similar colors if their intensities are similar
- The goal is to minimize the difference between the color $U(\mathbf{r})$ at pixel **r** and the weighted average of the colors at neighboring pixels

Objective function

$$J(U) = \sum_{\mathbf{r}} \left(U(\mathbf{r}) - \sum_{\mathbf{s} \in N(\mathbf{r})} w_{\mathbf{rs}} U(\mathbf{s}) \right)^2$$

color of pixel r

color of pixel s

sum over pixels in
the neighborhood of r

sum over all pixels

affinity between
r and s

$$J(U) = \sum_{\mathbf{r}} \left(U(\mathbf{r}) - \sum_{\mathbf{s} \in N(\mathbf{r})} w_{\mathbf{rs}} U(\mathbf{s}) \right)^2$$

Possible affinity functions:

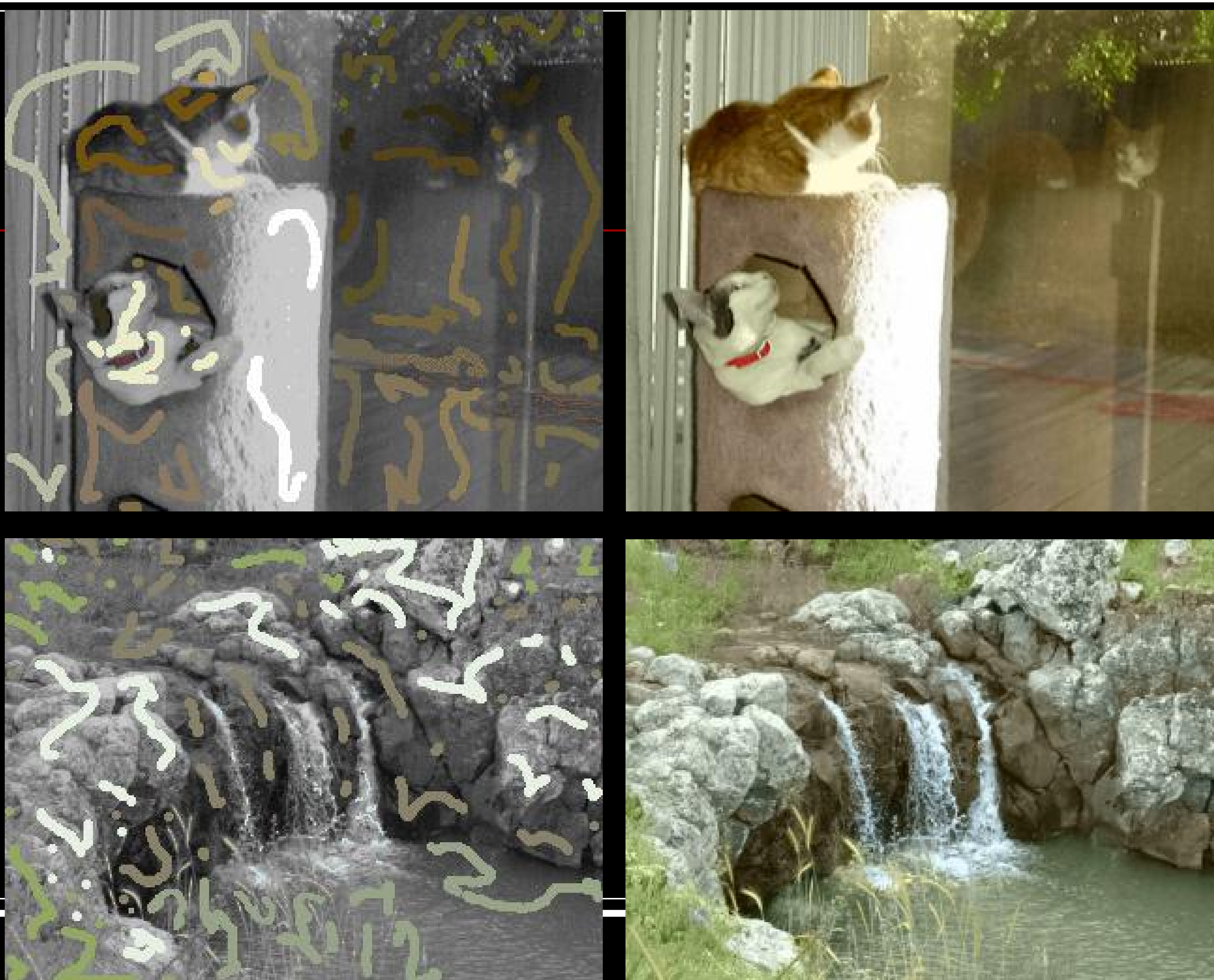
$$w_{\mathbf{rs}} \propto e^{-(Y(\mathbf{r})-Y(\mathbf{s}))^2/2\sigma_{\mathbf{r}}^2}$$

$$w_{\mathbf{rs}} \propto 1 + \frac{1}{\sigma_{\mathbf{r}}^2}(Y(\mathbf{r}) - \mu_{\mathbf{r}})(Y(\mathbf{s}) - \mu_{\mathbf{r}})$$

Neighborhood definition: for video, take *optical flow* into account

Constraints: color of user-specified pixels remains fixed

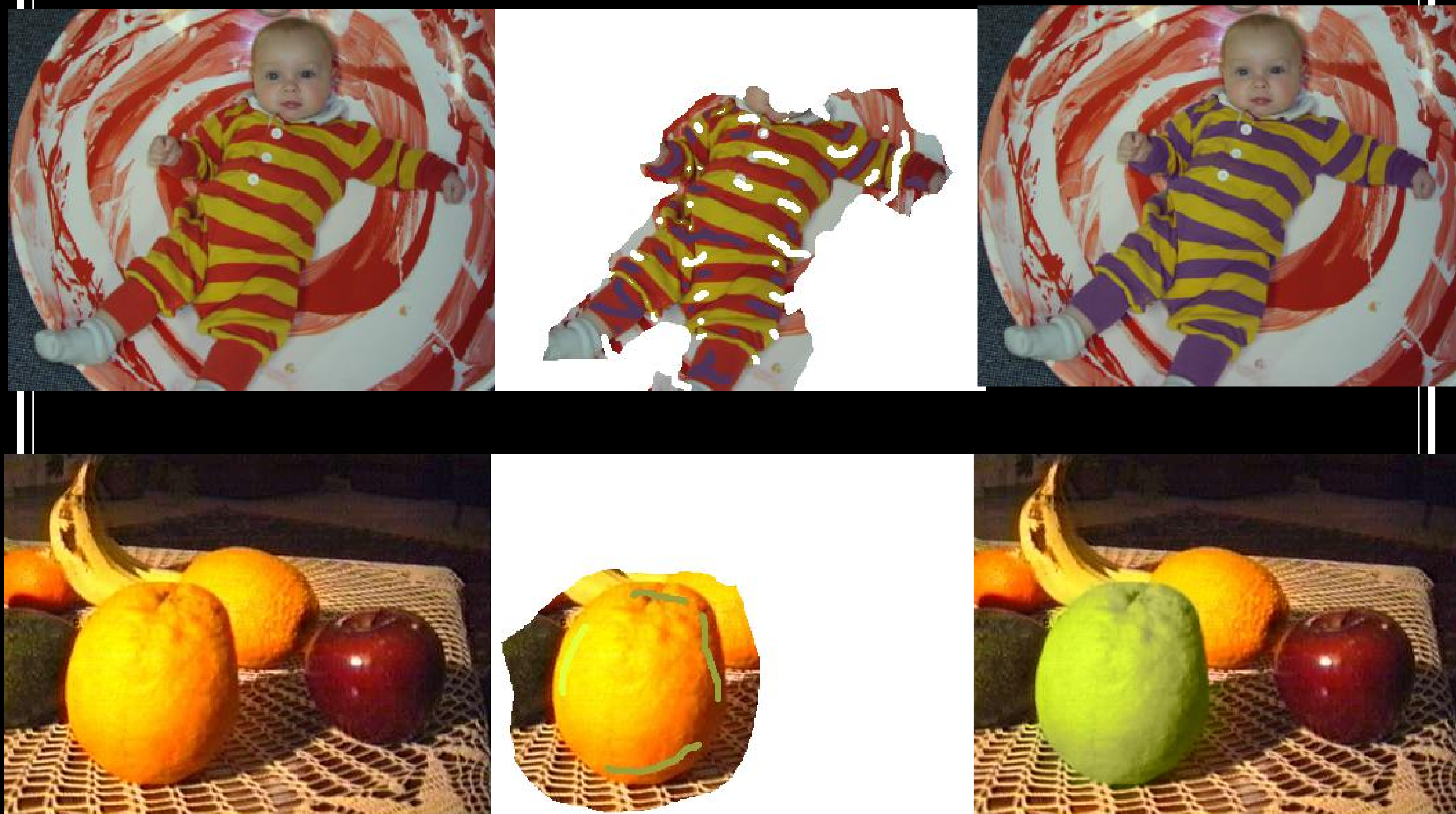
- Cost function is quadratic, and constrains are linear
- This optimization can be treated by solving a large sparse linear equations




CASINO
MONTE-CARLO

Comparison to segmentation-based colorization

Recoloring



More recoloring

Progressive colorization

Video colorization

Video colorization

Video colorization

Video colorization

Video colorization

Video colorization

DeColorization

- Color2Gray

Rgb2gray

- Gray = $\omega_r \mathbf{I}_r + \omega_g \mathbf{I}_g + \omega_b \mathbf{I}_b$
  - $\omega_r + \omega_g + \omega_b = 1$
  - In Matlab & OpenCV, ω_r = **0.299,** ω_g = **0.587,** ω_b = **0.114**

Traditional Methods

- Global optimization-based
  - Maximize differences between neighbor Pixels/Regions
  - Large-scale system

Problems

- Time-consuming
- Bad results

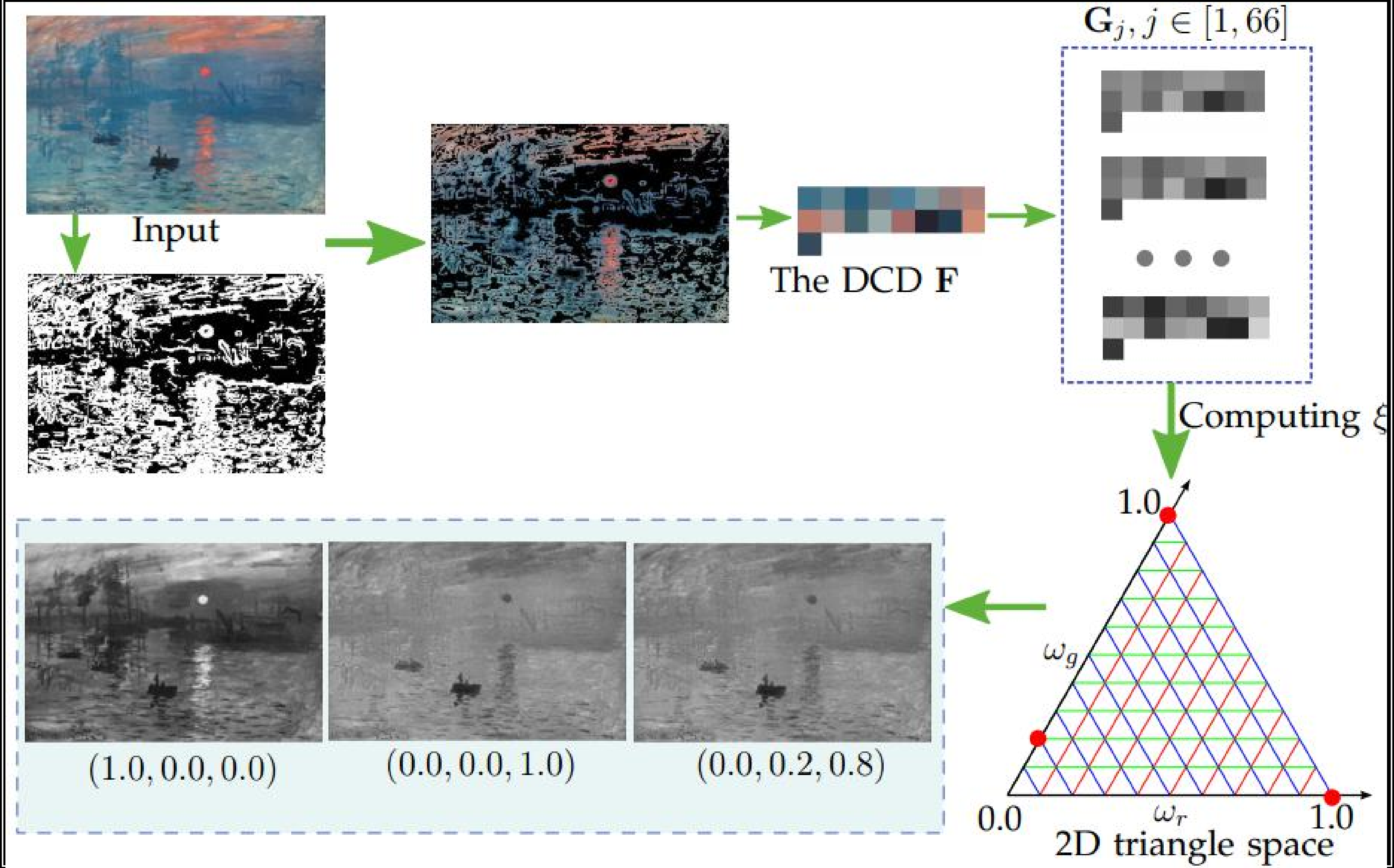
New Idea

- Dominant Color Hypothesis
  - Decolorized grayscale values of dominant colors around image edges directly reflect the quality of decolorization conversions
  - Small scale
  - Across whole image areas

Strategies

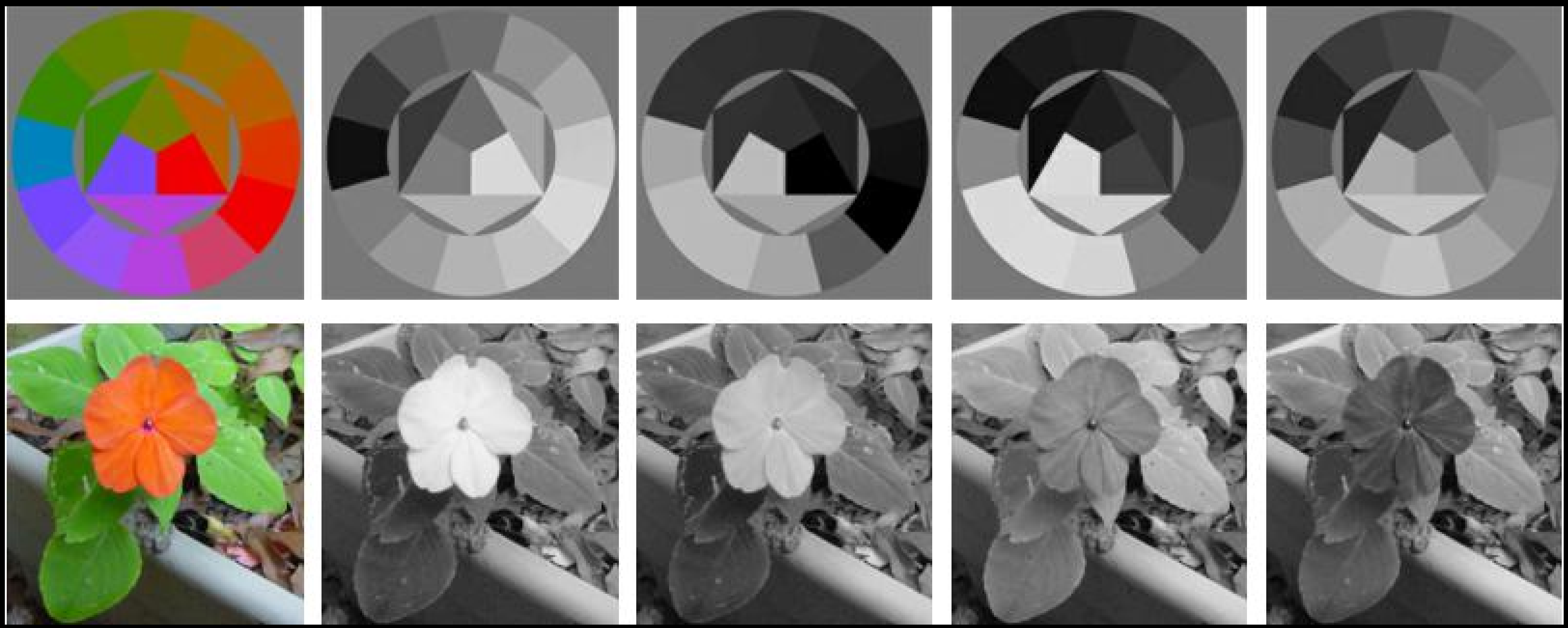
- Col_i -> $Gray_i$, $i = 0, 1, \ldots, N-1$
- $D_{i,j} = |Grayi – Grayj|$
- 3 strategies
  - Maximize the number of $D_{i,j}$ > threshold
  - Maximize the sum of Di,j
  - The more important of Col_i and Col_j, the more necessary to keep large $D_{i,j}$

- ω_r, ω_g, and ω_b are distributed in the range of [0, 1] with a searching interval of 0.1, and the sum is equal to one; therefore, there are total 66 possible sets of weights



Input
The DCD $\mathbf{F}$
$\mathbf{G}_j, j \in [1, 66]$
Computing ξ
1.0
ω_g
0.0
ω_r
1.0
2D triangle space
$(1.0, 0.0, 0.0)$
$(0.0, 0.0, 1.0)$
$(0.0, 0.2, 0.8)$

Ordered Results

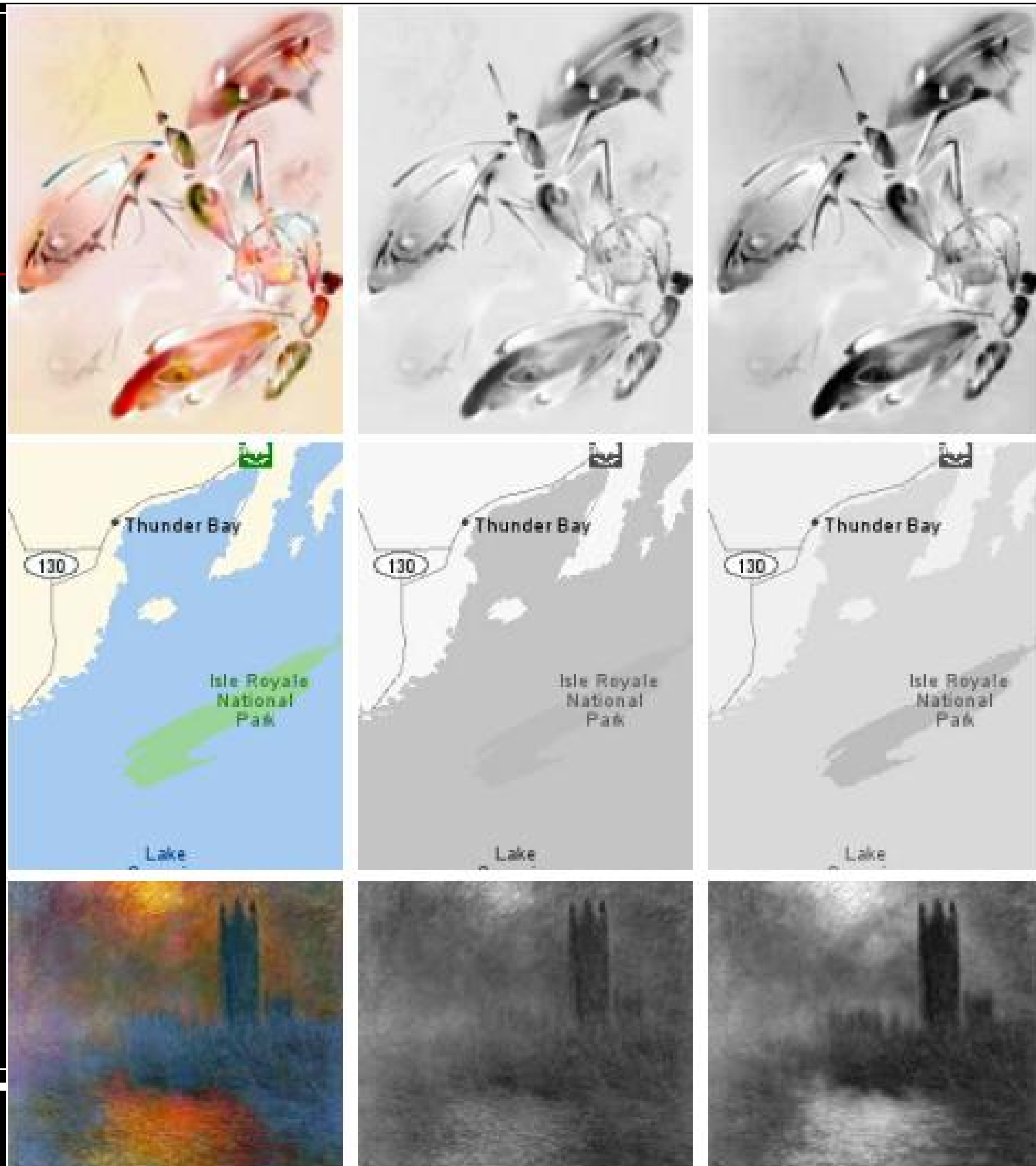


Performance

- $O(M \cdot N^2)$
  - M is the number of weights, fixed 66
  - N is the number of dominant colors, usually less than 100
  - So, $O(1)$ in fact

(b) 3200 × 2000, 21 dominant colors, 27 ms

(d) 1920 × 1280, 24 dominant colors, 26 ms


Thunder Bay
130
Isle Royale
National
Park
Lake