CSE 659A: Advances in Computer Vision

Spring 2020: T-R: 1:00-2:20pm @ Zoom

Instructor: Ayan Chakrabarti (ayan@wustl.edu).

http://www.cse.wustl.edu/~ayan/courses/cse659a/

Mar 24, 2020
HW 3 due tonight. HW 4 posted. Due April 9th.

Make sure you've looked at the new policies and timeline on course website.

Recorded videos of Zoom sessions available in Canvas (Zoom tab -> Recordings).
LAST TIME: DEHAZING

- Simplified observation model:

\[ Y[n] = t[n]L[n] + (1 - t[n])A \]

- \( L[n] \) is the haze-free image. \( Y[n] \) is the observation. Both are 3-channel.

- \( t[n] \) is a transmission coefficient that is dependent on depth. Assume \( t[n] = \exp(-\beta Z[n]) \).

- \( A \) is a common 3-channel vector for the entire image, called "Airlight". Color of the haze (depends on the light color).

- Estimate \( L[n] \) given \( Y[n] \). Ill-posed because you don't know \( t[n] \) or \( A \).

- Assume \( t[n] \) is constant within a patch, and someone gives you a pair of patches \( P_1[n] \) and \( P_2[n] \) (cropped from \( Y[n] \)) that you know are identical (in \( L[n] \))

\[
\begin{align*}
P_1[n] &= t_1 L[n] + (1 - t_1)A \\
P_2[n] &= t_2 L[n] + (1 - t_2)A \\
P_2[n] - A &= \frac{t_2}{t_1} (P_1[n] - A)
\end{align*}
\]

- Can get \( t_2/t_1 \) as ratio of standard deviations of pixel intensity inside \( P_1 \) and \( P_2 \).

- Can solve for \( A \)
DEHAZING

\[ Y[n] = t[n]L[n] + (1 - t[n])A \]
\[ P_1[n] = t_1 L[n] + (1 - t_1)A \]
\[ P_2[n] = t_2 L[n] + (1 - t_2)A \]

- Given a pair of patches that you know will match, you can compute \( t_2/t_1 \) and \( A \).
- But you need to find matching patch pairs.
- Realize that \( (P[n] - \text{Mean}(P[n]))/\text{std}(P[n]) \) depends only on \( L \).
- Find matches as euclidean distance between normalized patches.
- Get a set of matches, to get various estimates of \( A \) and pairwise constraints on \( t[n] \).
- Put all of this together into an energy term, add constraints that \( t[n] < 1 \) and \( t[n] > LB \), and minimize.
DEHAZING

Input

Our
DEHAZING
DEHAZING
DEHAZING
We said that similar patches recur in an image. This is typically because of repeating patterns. But there is also some local variation. Can we remove these variations to make the image look more 'regular'. Can we amplify them to better visualize small deviations in texture or patterns.
Basic Idea

- Extract set of patches from the image.
- Build a 'nearest neighbor field': for each patch, find a different nearest neighbor patch in the same image.
- Find the "difference" between a patch and it's nearest neighbor.
  - Fit a transformation that maps the patch to it's nearest neighbor.
- This transformation can be a color transformation or a flow-field (warp the nearest neighbor to the patch).
- Amplify, or attenuate this transformation (make it closer to the identity).
- Replace patch by its transformed version.
AMPLIFYING IRREGULARITIES

- Warp-based transforms
AMPLIFYING IRREGULARITIES

- Warp-based transforms
AMPLIFYING IRREGULARITIES

- Color-based transforms

(a) Input Image  (b) Color Exaggeration

(c) Reference Patch & NNs  (d) Color Correction
The idea of nearest neighbors and fitting transformations can be applied to other settings.

Shih et al., Data-driven Hallucination of Different Times of Day from a Single Outdoor Photo, SIGGRAPH Asia 2013.

• The goal, given an image of a scene taken in the day, hallucinate what it would look like at night. (or vice-versa)
• You are given the input image and the time of day it was taken

as well as a "time-lapse" database: images of other scenes, for each scene multiple images taken at different times of day.
HALLUCINATE LIGHTING CHANGES

Shih et al., Data-driven Hallucination of Different Times of Day from a Single Outdoor Photo, SIGGRAPH Asia 2013.

Basic Idea:

- You are given an image $X$ at time $t_i$ and want to output what it would look like at time $t_o$.
- You are also given a dataset of image pairs $(Y^j_i, Y^j_o)$ of different scenes at those times.
- Find a an image $Y_{p:i}$ from among $\{ Y^j_i \}$ similar to $X$ based on global matching features.
- Find a dense nearest neighbor field between $X$ and $Y_{p:i}$.
- Warm $Y_{p:i}$ to match $X$ based on this field. Also warp the frame of the matched scene at time $t_o$.
- Find the color transform between each pixel in the warped frames at time $t_i$ to time $t_o$.
- Apply this color transform to $X$. 
HALLUCINATE LIGHTING CHANGES

Shih et al., Data-driven Hallucination of Different Times of Day from a Single Outdoor Photo, SIGGRAPH Asia 2013.

(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).
HALLUCINATE LIGHTING CHANGES

Shih et al., Data-driven Hallucination of Different Times of Day from a Single Outdoor Photo, SIGGRAPH Asia 2013.
Shih et al., Data-driven Hallucination of Different Times of Day from a Single Outdoor Photo, SIGGRAPH Asia 2013.
In 559A, we talked about how shape, lighting, and reflectance affect image appearance.

For the lambertian case,

\[ I = \rho \langle \hat{n}, \ell \rangle \]

- \( I \) is the image of a surface point. \( \rho \) is the albedo, \( \hat{n} \) is a unit 3-vector giving the surface normal, and \( \ell \in \mathbb{R}^3 \) is the light: direction + magnitude giving us brightness.

- We talked about how if you take three different images with three different lights \( \ell \), you can use this to recover surface normals \( \hat{n} \) and albedo \( \rho \).

- But this is in a lab setting with calibrated lighting. How much does shading and photometric effects tell us about shape, "in the wild"?

- One way to analyze this is by looking at uniqueness. Are there multiple shapes that, combined with their own lights, would give us the same image?
PHOTOMETRIC IMAGE ANALYSIS

• Now we know that a single observation $I = \rho \langle \hat{n}, \ell \rangle$, does not uniquely determine $\hat{n}$ even when $\ell$ and $I$ are known.

Why not?

• In the calibrated setting, $I$ gives us the "angle" between the surface normal and the light in 3D. But this still limits the surface normal to a cone of directions.

• But we are not analyzing each pixel in isolation. We know that the normal gives us the derivative of depth, and therefore the 'normal' field needs to be integrable.

• What is the ambiguity once we take integrability as a constraint?

• For simplicity, assume orthographic projection: $(x, y, z)$ projects to $(x, y)$ in the image.
Assume object has depth as some function $z(x, y)$ of 2D co-ordinates.

Therefore, the surface of the object is the set of points $(x, y, z(x, y))$.

The surface normal at point $(x, y)$ is given by

$$\hat{n}(x, y) \propto \begin{bmatrix} -z_x(x, y) \\ -z_y(x, y) \\ 1 \end{bmatrix}$$

where $z_x = \partial z(x, y)/\partial x, z_y = \partial z(x, y)/\partial y$.

The $\propto$ implies the surface normal must be normalized to be unit length.

We can now reason about different $z(x, y)$ that give us the same image, because their normal fields will automatically be integrable.
The Bas-Relief ambiguity gets its name from bas-relief sculptures.

- These are shallow sculptures carved on surfaces which give the appearance of depth.
- This is because there is an inherent ambiguity between $z(x, y)$ and a Bas-Relief transformed version $z'(x, y)$. They will look the same if we allow for a related transformation of the light and albedo.
- It's obvious that modifying the albedo will give us equivalent shading. But this is true also for shadows!
PHOTOMETRIC IMAGE ANALYSIS

- Surface is \( z(x, y) \), surface points \( p(x, y) = (x, y, z(x, y)) \).

\[
\hat{n}(x, y) \propto \begin{bmatrix} -z_x(x, y) \\ -z_y(x, y) \\ 1 \end{bmatrix}
\]

- Consider the family of GBR transformations for arbitrary scalars \( \lambda, \mu, \nu \):

\[
z'(x, y) = \lambda z(x, y) + \mu x + \nu y
\]

\[
p' = Gp, \quad G = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \mu & \nu & \lambda \end{bmatrix}, \quad G^{-1} = \frac{1}{\lambda} \begin{bmatrix} \lambda & 0 & 0 \\ 0 & \lambda & 0 \\ -\mu & -\nu & 1 \end{bmatrix}
\]

\[
z'_x(x, y) = \lambda z_x(x, y) - \mu, \quad z'_y(x, y) = \lambda z_y(x, y) - \nu
\]

\[
\hat{n}'(x, y) \propto G^{-T} \hat{n}(x, y)
\]

Claim: If I observe \{p\} under light \( \ell \), and \{p' = Gp\} under light \( G\ell \): The shadows will be in the same place.
Two kinds of shadows:

- Attached Shadows: $\ell^T \hat{n} < 0$.

$$\ell'^T \hat{n}' \propto (G\ell)^T (G^{-T} \hat{n}) = \ell^T G^T G^{-T} \hat{n} = \ell^T \hat{n}$$

So if $\ell^T \hat{n} < 0$, so will $\ell'^T \hat{n}'$.

- Cast Shadows: One surface point blocks the light to another.

Can show these will stay the same by reasoning about shadow boundaries.
Let's say you have point $p_1$ and $p_2$ are boundaries of the shadow on a surface, where $p_2$ (in attached shadow) is blocking light to point $p_1$ (in a cast shadow).

- We can show that in the transformed surface, $p_2'$ will also block light to $p_1'$.

Conditions for a shadow boundary pair:

$$n_2^T \ell = 0, \quad p_2 - p_1 = \gamma \ell$$

$$n_2'^T \ell' \propto (G^{-T}n_2)^T (G\ell) = 0$$

$$p_2' - p_1' = G(p_2 - p_1) = G(\gamma \ell) = \gamma G\ell = \gamma \ell'$$
So cast and attached shadows will match.

This means that if I change the albedo of the surface at each point, so that $\rho' \hat{n}'T \ell' = \rho \hat{n}T \ell$, then the two images will look exactly the same. And sometimes for human perception, the albedo change is not even necessary.