Administrivia

Assignment two is now posted:
http://www.cs.arizona.edu/classes/cs477/spring10/ua_cs_only/assignments
Slides now being posted:
http://www.cs.arizona.edu/classes/cs477/spring10/ua_cs_only/lectures
Lectures and assignments will require either connecting from a UA machine, OR
login id ("me") and password ("vision4fun").

TA office hours (in 927B)
Mondays 10-11am
Wednesdays 1-2pm
Kobus’s office hours by electronic sign up (reserve by 6:00pm previous day)
Tuesdays at 5-5:20 pm
Thursdays 9-9:30 am AND 5-5:20 pm

Image Formation (non-linear transform)

\( f(v) \) is often ignored (assumed to be the identity), but this is not a safe assumption, especially when color or radiometric measurements matter.

Commonly images are “gamma” corrected by raising the RGB values (normalized to \([0,1]\)) to the power \(1/(2.2)\).

Note that in such an image, a number twice as large does not mean that the light had twice the power!

To linearize RGB’s from such a signal we compute:

\[
p = F^{-1}(v) = 255^*(v/255)^{2.2}
\]

Image Formation (non-linear transform)

The non-linear transformation is added by captured devices after the raw capture (which is typically linear).

Because it is a single function applied to responses, it is easy to measure and compensate for.

Image Formation (non-linear transform)

Why are images typically encoded in this way?

Historically, images have been gamma corrected on the assumption that their values drive a CRT (cathode ray tube) monitor which are non-linear devices. Their theoretical response to a voltage is energy output proportional to that voltage raised to the \((5/2)\) power. Appropriately gamma corrected images display as linear on such devices.
**Image Formation (non-linear transform)**

Coincidentally, this typically gamma correction is a sensible way to encode image data into a limited number of values (e.g. 256) due to the noise sensitivity of the human vision system.

Hence, while CRT displays are now obsolete, images are still typically non-linear, and the signal to modern displays (which are linear) are typically adjusted assuming typical incoming non-linear in images.

If you have access to a Mac, then you can play with this under System Preferences --> Displays --> Color --> Calibrate

---

**Linear Least Squares (§3.1)**

- Very common problem in vision: solve an over-constrained system of linear equations
  - e.g., $Ux = y$, where $U$ has more rows than needed
  - e.g., $Ux = 0$, $|x| = 1$, where $U$ has more rows than needed
- More equations allows multiple measurements to be used
- Least squares means that you minimize squared error (the difference between your model and your data)
- Least squares minimization is (relatively) easy
- Not very robust to outliers (assumes error is Gaussian)

We will look at two problems

First, $Ux = y$ where $U$ has more rows than needed

Second, $Ux = 0$ subject to $|x| = 1$ were $U$ has more rows than needed

We can use the first for naïve spectral camera calibration.

We will use the second problem for geometric camera calibration.

---

**Non-homogeneous Least Squares**

Problem one $Ux = y$ where $U$ has more rows than needed

$U$ is not square, so inverting it does not work

In fact, usually there is no solution. We need to redefine what it means to “solve the equation”.

We seek the “best” answer but what is that?

* This is regression by a different name.
Non-homogeneous Least Squares

Define \( e = Ux - y \) and \( E = \| e \|^2 = e^T e \)

The least squares solution which is the one that has minimum \( E \).

We can derive the answer by differentiating with respect to each \( x_i \), and setting all resulting equations to zero (see supplementary slides).

The answer is given by
\[
x = U^T y \quad \text{where} \quad U^T = (U^T U)^{-1} U^T \text{ is the pseudoinverse of } U
\]

Non-homogeneous linear least squares summary

(important)

Non-homogeneous linear least squares

You should be able to set up
\[
Ux = y
\]

You should know that it is solved by
\[
x = U^T y \quad \text{where} \quad U^T \text{ is the pseudoinverse of } U
\]

You can assume that you can look up
\[
U^T = (U^T U)^{-1} U^T
\]

*You should also keep in mind that for numerical stability, one may have to use a different approach to solve (without matrix inversion) the following
\[
U^T U x = U^T y
\]

Non-homogeneous linear least squares

(important)

(important)

(important)

Non-homogeneous linear least squares

(important)

(important)

(important)

(important)

(important)

Strategy: measure some spectra entering the camera, \( L_i \), and note the response, \( \rho_i \).

So we have, for a bunch of measurements, \( i \):
\[
\rho_i = L_i \bullet R
\]

If we don’t have enough measurements, then the problem is under constrained. To account for noise, we want to use multiple measurements.
Non-homogeneous linear least squares (example one---naïve spectral camera calibration)

From:
\[ \rho_i = L_i \cdot R \]

The path is clear. Just form a matrix \( L \) with rows \( L_i \), a vector \( P \) with elements \( \rho_i \), and solve the least squares equation
\[ L \cdot R = P \]

Non-homogeneous linear least squares (example two---naïve line fitting)

Can write \( y = mx + b \) as:
\[ (x \ 1) \cdot (m \ b) = y \]

Important

Image Formation (Geometric)

Non-homogeneous linear least squares (example two---naïve line fitting)

Can write \( y = mx + b \) as:
\[ (x \ 1) \cdot (m \ b) = y \]

Important

Non-homogeneous linear least squares (example two---naïve line fitting)

Can write \( y = mx + b \) as:
\[ (x \ 1) \cdot (m \ b) = y \]

Important

Image Formation (Geometric)
Pinhole cameras

- Abstract camera model--box with a small hole in it
- Pinhole cameras work for deriving algorithms--a real camera needs a lens

Distant objects are smaller

Size Constancy

Object size vs. object depth

(Images copyright John H. Kranz, 1999)

(Slide courtesy Frank Dellaert)