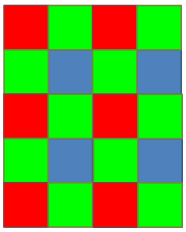
COSC470 Images and Filters

COSC470: Special Topic
Computer Vision | 3D Reconstruction
Steven Mills

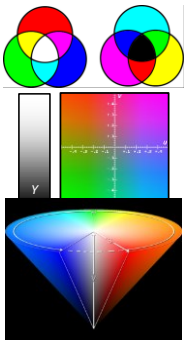
Digital Images

- Images are 2D arrays
- Each element (pixel) can have a vector of values
  - Single value (greyscale/intensity)
  - Colour (RGB common, also HSV, YUV)
- Capture and display use single value at each point



Representing Colour

- Red-Green-Blue
  - Relates to our eyes
  - Common for hardware
- YUV (or YCbCr)
  - Brightness + 2 colour axes (B-Y and R-G)
- Hue-Saturation-Value
  - More meaningful
- Cyan-Magenta-Yellow
  - Inverse of RGB for inks



Colour Conversion

- RGB ↔CMY

$$R = 1 - C \qquad C = 1 - R$$
$$G = 1 - M \qquad M = 1 - G$$
$$B = 1 - Y \qquad Y = 1 - B$$

- RGB ↔ YUV

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} \approx \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.147 & -0.289 & 0.436 \\ 0.615 & -0.515 & -0.100 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} \approx \begin{bmatrix} 1 & 0 & 1.140 \\ 1 & -0.395 & -0.581 \\ 1 & 2.032 & 0 \end{bmatrix} \begin{bmatrix} Y \\ U \\ V \end{bmatrix}$$

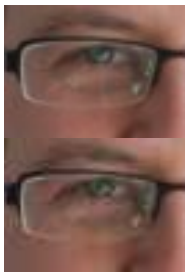
- RGB→HSV
- V and S fairly simple:

$$V = \max(R, G, B)$$
$$S = \frac{\max(R, G, B) - \min(R, G, B)}{\max(R, G, B)}$$

- H a little more complex, but basically interpolate between R, G, B on the colour wheel
- If $V = 0$, then $H = S = 0$

Image File Formats

- Compression
  - None (eg: PNM)
  - Lossless (eg: PNG)
  - Lossy (eg: JPEG)
- Colour representation
  - RGB (PNM)
  - YUV (JPEG uses Y'CrCb)
  - Lookup (some PNGs)
- Transparency



Images in OpenCV

- Basically a 2D array – the `cv::Mat` class
- A 2D array of what?
  - `double float, uchar, int`
  - `cv::Vec3b = cv::Vec<uchar, 3>`
  - `cv::Scalar = cv::Vec<float, 4>`
  - CV_8UC3, CV_32F, etc.
- Indexed how?
  - *x, y* or row, column?
  - Origin and axis directions?

Reading and Writing Images

- Using highgui library

```
cv::imread(filename, flag=1);
```

  - `flag`: whether to make a colour image (>0), greyscale (0), or use what is in the file (<0)

```
cv::imwrite(filename, img, params);
```

  - File type depends on `filename` extension
  - `params`: a `vector` of `int`s controlling parameters such as compression/quality level

Accessing Image Elements

- Can access the directly:

```
cv::Mat img(cv::Size(w,h),CV_8UC3);
img.at<cv::Vec3b>(i,j) = …
```

- But need to know the type for this
- Higher level functions make this easier

```
cv::circle(img, cv::Point(100,100),
           50, CV_RGB(255,0,0), 5);
```

Image Filtering

- Convolution filters are defined by a *kernel*
- Kernel is placed over each pixel in the image
- Pixel values and kernel values are multiplied
- New pixel value is the sum of these products

$$K = \frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}$$

$$\frac{1}{16}\begin{pmatrix}7\times1+4\times2+6\times1+\\ 9\times2+3\times4+7\times2+\\ 2\times1+4\times2+5\times1\end{pmatrix} = \frac{80}{16} = 5$$

$$I'(x,y) * K = \sum_{dx=-r}^{dx=r}\sum_{dy=-r}^{dy=r} I(x+dx, y+dy)K(dx+r, dy+r)$$

Some Common Filters

- Mean filters:

$$K_1 = \frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

$$K_2 = \frac{1}{25}\begin{bmatrix}1 & 1 & 1 & 1 & 1\\ 1 & 1 & 1 & 1 & 1\\ 1 & 1 & 1 & 1 & 1\\ 1 & 1 & 1 & 1 & 1\\ 1 & 1 & 1 & 1 & 1\end{bmatrix}$$

- Sobel filters:

$$K_x = \begin{bmatrix}-1 & 0 & 1\\ -2 & 0 & 2\\ -1 & 0 & 1\end{bmatrix}$$

$$K_y = \begin{bmatrix}-1 & -2 & -1\\ 0 & 0 & 0\\ 1 & 2 & 1\end{bmatrix}$$

- Gaussian filters:

$$G(x,y) = \frac{1}{2\pi\sigma^2}e^{-\frac{x^2+y^2}{2\sigma^2}}$$

- Eg: $\sigma = \sqrt{1/2}$:

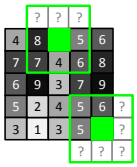
0.0008	0.0066	0.0133	0.0066	0.0008
0.0066	0.0547	0.1110	0.0547	0.0066
0.0133	0.1110	0.2251	0.1110	0.0133
0.0066	0.0547	0.1110	0.0547	0.0066
0.0008	0.0066	0.0133	0.0066	0.0008

- Truncating small values, scaling so that the corner values are 1 and rounding gives

$$K_{\sqrt{0.5}} \approx \frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}$$

Image Filtering Issues

- What happens at the edges of the image?



- Storing floating point values, negative, or out of range results?

$$K = \frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

$$K = \begin{bmatrix}-1 & -2 & -1\\ 0 & 0 & 0\\ 1 & 2 & 1\end{bmatrix}$$

$$K = \begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 2 & 1\end{bmatrix}$$

Separable Filters

- Filtering is expensive
  - A square filter of radius r requires $O(r^2)$ operations per pixel
- Many 2D filters can be split into two 1D filters
  - This requires only $O(r)$ operations
- Can rewrite kernels as product of two vectors

- 3x3 Mean filter:

$$K_1 = \frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix} = \begin{bmatrix}1/3\\ 1/3\\ 1/3\end{bmatrix}\begin{bmatrix}1/3 & 1/3 & 1/3\end{bmatrix}$$

- Sobel filter in x:

$$K_x = \begin{bmatrix}-1 & 0 & 1\\ -2 & 0 & 2\\ -1 & 0 & 1\end{bmatrix} = \begin{bmatrix}1\\ 2\\ 1\end{bmatrix}\begin{bmatrix}-1 & 0 & 1\end{bmatrix}$$

- Gaussian filter:

$$G(x,y) = \frac{1}{2\pi\sigma^2}e^{-\frac{x^2+y^2}{2\sigma^2}} = \left(\frac{1}{\sigma\sqrt{2\pi}}e^{\frac{-x^2}{2\sigma^2}}\right)\left(\frac{1}{\sigma\sqrt{2\pi}}e^{\frac{-y^2}{2\sigma^2}}\right)$$

Other Filters

- Not all filters are separable
- *Difference of Gaussians* $K_{\sigma_1} - K_{\sigma_2}$ is not, even though the parts are
- The *Laplacian* is not

$$L(x,y) = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}$$

$$K_L \approx \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

- Not all filters can be implemented using convolution kernels
- *Median filter* replaces each pixel by the middle value from a local area
- *Anisotropic diffusion* is a smoothing method that preserves edges

Filtering in OpenCV

- Specific functions for common filters

```
cv::GaussianBlur(), cv::blur(),
cv::medianBlur(), cv::Sobel()
```

- Generic functions for user-defined filters

```
cv::filter2D(src, dst, ddepth, kernel)
cv::sepFilter2D(src, dst, ddepth,
    rowKernel, columnKernel);
```