Computer Vision

91.450 Robotics I
Prof. Yanco
Spring 2015
RGB Color Space

- Lighting impacts color values!
HSV Color Space

- **Hue**, the color type (such as red, blue, or yellow);
  - Measured in values of 0-360 by the central tendency of its wavelength

- **Saturation**, the 'intensity' of the color (or how much grayness is present),
  - Measured in values of 0-100% by the amplitude of the wavelength

- **Value**, the brightness of the color.
  - Measured in values of 0-100% by the spread of the wavelength
Looking for Colors

• Can train on colors in a region of the image, then track that color

• Best to track colors in HSV color space (RGB is too lighting dependent)
Image Processing Pipeline

- Grab image
- Filter to smooth image
- Process for some property
  - Intensity changes for edges
  - Blobbing to find an area of a particular color
- Act on the results
Filtering Methods

- Median
- Mean
- Gaussian
Gaussian filter

- This filter $H$ is a good approximation to $h(u, v) = \frac{1}{2\pi\sigma^2}e^{-\frac{u^2+v^2}{2\sigma^2}}$
- Properties of Gaussian
  - more weight to the center
  - good model of blurring in optical systems
  - $\sigma$ corresponds to width of the Gaussian
Gaussian Filter
Mean Blur

• Blurs the image by changing the color of the pixel being looked at to the mean value of the pixels surrounding it. The number of surrounding pixels being looked at is defined by the kernel parameter. If kernel is 3, then the pixel being looked at is the center of a 3x3 box, shown in the diagram.
Mean Blur
Median Blur

- Blurs the image by changing the color of the pixel being looked at to the median value of the pixels surrounding it. The number of surrounding pixels being looked at is defined by the kernel parameter. If kernel is 3, then the pixel being looked at is the center of a 3x3 box, shown in the diagram.

![Median Blur Diagram](image)
Median Blur
Edge Detection: Sobel

\[
\begin{align*}
|G| &= \sqrt{G_{x}^{2} + G_{y}^{2}}
\end{align*}
\]
Edge Detection: Sobel
Edge Detection: Canny

1. Apply Gaussian filter
2. Sobel edge detection
3. Find direction of edges
4. Relate edge direction to direction that can be traced in an image
5. Nonmaximum suppression used to trace along the edge in the edge direction to suppress any pixel value that is not considered to be an edge
6. Hysteresis used to eliminate streaking (breaking up of an edge contour)
Edge Detection: Canny
Color Filtering
Stereo Vision

Finding a 3D point cloud from a pair of 2D stereo images via stereo matching and disparity calculations.

From Brigit Scroeder
Uses of Computer Vision: Surgical Systems
Uses of Computer Vision: Content Based Image Retrieval

Sample Image

Retrieved Images
Uses of Computer Vision:
Face Detection
Uses of Computer Vision: Street Crossing

- Tracking from a moving platform
- Need to look left and right to find a safe time to cross
- Need to look ahead to drive to other side of road
- Must stay in crosswalk
Algorithm for Tracking Cars

- Image differencing to find motion
- Noise filtering using 3x3 median filter
- Computation of Sobel edges
- Use Mori’s sign pattern to find bottoms of cars [Mori 1994]
- Find bounding boxes of moving objects
- Use knowledge from prior frames to mark direction of travel of each bounding box
Where art thou, red ball?

- Horswill 1997
Vision Setup

- The USB cameras in the Botball kit will work in either of the CBC’s USB ports
- Plug in a camera and you will be able to see the camera image by going to the vision screen
  - If you unplug the camera, the CBC may no longer recognize it if you plug it back in
    - You will need to restart the CBC if this happens

From KIPR
HSV Color Selection Plane

Hue=0
Sat=0
Val=224

Hue=360

Sat=224
Val=0

Note: 224 is the range of values the camera pixels put out in each of R, G & B

From KIPR
Color Blobs

- Each pixel on the screen has an HSV color
- When we say red, we really mean a range of HSV colors on the color selection plane that are approximately red
- A rectangular piece of the color selection plane that corresponds to being red specifies the range of HSV colors to be viewed as red by the CBC
  - This is called an HSV color model
- A red blob is all contiguous pixels matching one of the HSV colors in the red range
- A blob has a size, position, number of pixels, major and minor axis, etc.
Vision System Color Models

• The CBC can handle 4 HSV color models simultaneously
• It can track a number of blobs from each color model
• It can display the video in any one of three ways
  – **Raw** (live video)
  – **Match** (pixels matching the color model are highlighted)
  – **Tracked** (highlights matching pixels and shows blob boundaries and centroids)

From KIPR
Color Vision Interface

Vision..Tracking Screen

- **Raw** image is displayed

- Color Model 0 is being manipulated

- The **Bottom Right** corner of the color selection box is being adjusted

- It can be moved **Left, Right, Up or Down**
Color Vision Interface

Vision..Tracking Screen

- **Matched** image is being displayed
- Pixels that correspond to selected color region are shown highlighted

From KIPR
Color Vision Interface

*Vision..Tracking Screen*

- **Tracked** image is being displayed
- The bounding boxes of the tracked blobs are displayed, along with the centroid of each blob

From KIPR
Training a Color Channel

• Any color channel (in Match or Tracked mode) can be trained for tracking color blobs that match a given HSV color model by using the Vision..Tracking screen

• The default settings for color model 0 are for pixels that are approximately red, yellow for model 1, green for model 2, and blue for model 3

• Once set, the vision settings from training are retained
  – Default settings can be restored from the CBC Settings screen
  – Hint: if you are using the camera, shrink the selection box as small as possible for any color channels you are not using (reduces processing load)

From KIPR
Vision System Library Functions

• The CBC library function `track_update();` is a command that causes the CBC to capture the most recent camera frame for analysis
  – Frame analysis determines blob properties such as the (x,y) coordinates of the centroid of the blob

• `track_count(3);` provides how many (blue) blobs are being seen for the model 3 track
  – If the count is 0 there are no (blue) blobs
  – Blobs are numbered 0,1,2, … from largest to smallest

• `track_x(3,0);` for track 3, blob 0, returns the value of the x coordinate of the centroid of the largest (blue) blob

From KIPR
Image Coordinates

- The camera’s processed field of view is treated as an x-y (column,row) coordinate array
  - The upper left corner has coordinates (0,0)
  - The lower right corner has coordinates (159,119)
  - The CBC display does not show the camera’s full field of view

From KIPR
Example Using Vision Functions

// Train the camera so that it recognizes a red colored
// object for color channel 0
int main() {
    int x, y, color=0;  // set up for color channel 0 (red)
    while (black_button() == 0)  //run till button is pressed
    {
        track_update();  // process the most recent image
        if (track_count(color) > 0)
        {
            // get x, y for the biggest blob the channel sees
            x = track_x(color,0); y = track_y(color,0);
            printf("Biggest blob at (%d,%d)\n",x,y);
        }
        else
        {
            printf("No color match in Frame\n"); }
        sleep(0.2);  // give print time to register
    }
    printf("Program is done.\n");
}