next wednesday (Sept. 27):
1-2pm: bring your touchpad to class, we grade step 1/2.
2-3pm: Jared helps you fixed step 1/2.
(no lecture, I have to be at a event in Germany)

if there’s time, Jared can help with 3D printing (HW6) and OpenCV installation (HW7).

final grading: live demo with your application after you submitted step 1-4.

resistor and circuit wiring.

6.819/6.869: Advances in Computer Vision
computer vision is a huge topic, we will only cover a few intro level things to get you started

6.819/6.869: Advances in Computer Vision
today: preparation for the second part of the pset
segmentation:
locating object in an image
reading a live camera image
camera_port = 0
camera = cv2.VideoCapture(camera_port)
disclaimer:
code might vary
depending on which OpenCV version you use

we recommend OpenCV for python.
image color spaces
the **two color space** people typically know:

- For **digital display**:

  ![RGB](image1)

- For **printed content**:

  ![CMYK](image2)
the camera image you get will be RGB
how to find fingerprint?

- identify regions with a specific color (LED color)
- ignore everything else

fingerprints appear brightener and have a specific color since the LED light bounces off from them
is RGB a good color space for tracking color?

<30s brainstorming>
RGB color space:

R: red channel
G: green channel
B: blue channel

3 channels for color

![Color Picker](image)
if light conditions change, how to know how to adjust the values?
HSV color space::

H: hue (aka color)  (0-255)
S: saturation        (0-255)
V: value / brightness (0-255)
so for which H,S,V values do you want to look in your image?

<30s brainstorming>
H: hue (aka color)  (0-255)
S: saturation       (0-255)
V: value / brightness (0-255)

so for which H,S,V values do you want to look in your image?

- Hue: your specific color, e.g. 60-140 for green
- Saturation: 0-255 (all saturated versions of your color!)
- Value: 0-255 (all bright and not so bright versions of your color)
convert RGB image to HSV

```python
hsv_image = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)
```
the colors after conversion to HSV look weird…
any idea why?

<30s brainstorming>
the colors after conversion to HSV look weird… any idea why?

- HSV cannot be displayed on an RGB display!
- The colors you see here is not what’s represented internally.
thresholding an image
thresholding an image:

- convert everything with a specific value to white.
- convert the rest to black
- black / white image = binary image
thresholding image:

lower_blue = np.array([h_min, s_min, v_min])
upper_blue = np.array([h_max, s_max, v_max])

threshold_image = cv2.inRange(hsv_image, lower_blue, upper_blue)
if you couldn’t call OpenCV’s threshold function how would you implement it yourself?

<30s brainstorming>
if you couldn’t call OpenCV’s threshold function how would you implement it yourself?

for each pixel access value
if value > X make white
if value < X make black
create some sliders to make it easier to find the right values
thresholding can be super easy or very difficult…
let’s consider this image

here’s how the histogram works

black  white
let's consider this image

where would you place the threshold to find all trees?

<30s brainstorming>

black white
let's consider this image

where would you place the threshold to find all trees?

black
white
here it’s much harder : /
this is why we gave you colored LEDs...
gives you a nice histogram with ‘all fingers in one bucket’

this is what inRange() does:
ways to segment
when **object and background are too similar**...
background subtraction:

- object = frame with object - clear frame
histogram back projection:

- ask the user for a ‘sample’ of the object
- e.g. ‘hold your hand into the middle of the camera’

use all these grey values for thresholding
depth camera:

- in the image, things in the front are rendered brighter
- things in the back darker
- easy to threshold
find contours
so when do two white pixels A and B belong to the same object?
so when do two white pixels A and B belong to the same object? if there is a path connecting them along which all pixels are white
connected components:

CC = 0
Scan across rows:
If 1 and connected:
  Propgate lowest label
  behind or above
  (4 or 8 connected)
  Remember conflicts
If 1 and not connected:
  CC++ and label CC
If 0:
  Label 0
Relabel based on table
connected components:

CC = 0
Scan across rows:
  If 1 and connected:
    Propgate lowest label
    behind or above
    (4 or 8 connected)
    Remember conflicts
  If 1 and not connected:
    CC++ and label CC
  If 0:
    Label 0
    Relabel based on table

where are the conflicts?

<30s brainstorming>
connected components
= find contours:
contours, hierarchy = cv2.findContours(threshold_image,
cv2.RETR_TREE,
cv2.CHAIN_APPROX_SIMPLE)
find contours:

contours, hierarchy = cv2.findContours(threshold_image, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)

contour hierarchy information
(RETR_LIST = all contours, no hierarchy information)
(RETR_TREE = which contour is inside which other etc.)
only consider contours of a certain size:

```python
for contour in contours:
    area = cv2.contourArea(contour)
    if area > 1000:
        cv2.drawContours(frame, contour, -1, (0,255,0), 3)
```

Hierarchy level to be drawn
-1 = all hierarchies (all contours)
2 = only hierarchy 2

color thickness

add a slider for the **finger area size**!
contours
center of a contour
find center of a contour:

```python
cv2.minEnclosingCircle(contour)
```

contour = path of points

fit circle = center, radius
drawing shapes and text onto an image
cv2.circle(frame, draw_center, draw_radius, (0, 255, 0), 2)
cv2.putText(frame, textstring, (int(x+50), int(y)),
            cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255, 255, 255), 1)
end.