Name: Joshua Lee 11-10-2025

# Computer Vision Rover

1) Submit the screenshot verifying that Experiment 2 worked. What were the 6 numbers you used for detecting the color orange?

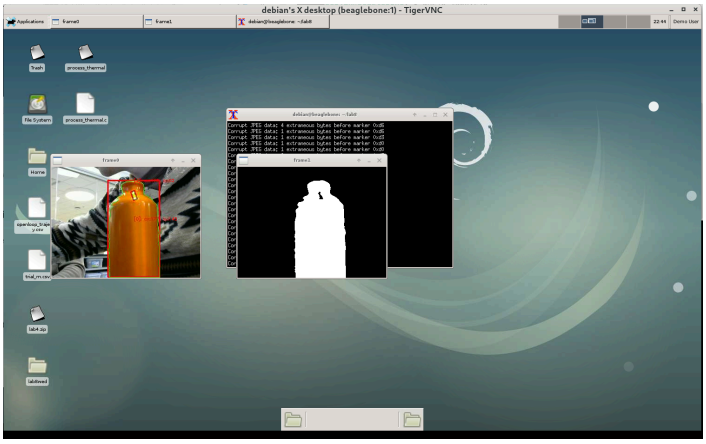


*Figure 1. Orange water bottle detected in experiment 2*

orange_HSV_min = (3, 90, 90)

orange_HSV_max = (25, 255, 255)

2) Submit your code for Experiment 3, just the code just the code for center_orange() and forward_to_orange(), along with the lab instructor's initials verifying that exp3 worked

```
printf("----------START center_orange----------\n");
// turn until "orange" object is seen with camera:
status = turn_until(turn_increment*DEG2RAD,ORANGE);
camera_process_next_frame();
print_blob();
done=0;
while(!done){
    // if blob exists and has enough pixels: turn towards it
    // otherwise: keep turning
    // students fill in code here....
    if(is_orange_visible()) {
    recenter_orange();
    } else {
    turn(5*DEG2RAD);
    }

    camera_process_next_frame();
    print_blob();
    // done with while loop if:
    //   1. orange blob is centered in the camera,
    //   2. the turn increment is really small
    if(is_orange_visible_centered()) {
        printf("center: DONE: orange_centered\n");
        done=1;
    }
    if(fabs(turn_increment) < blob_locationthresh*0.1) {
        printf("center: DONE: turn_increment is small: %f < %f\n",
                fabs(turn_increment), blob_locationthresh*0.1);
        done=1;
^G Get Help  ^O Write Out  ^W Where Is  ^K Cut Text  ^J Justify
^X Exit      ^R Read File  ^\ Replace   ^U Uncut Text ^T To Spell
```

```
status = turn_until(turn_increment*DEG2RAD,ORANGE);
camera_process_next_frame();
print_blob();
done=0;
while(!done){
    // if blob exists and has enough pixels: turn towards it
    // otherwise: keep turning
    // students fill in code here....
    if(is_orange_visible()) {
    recenter_orange();
    } else {
    turn(5*DEG2RAD);
    }

    camera_process_next_frame();
    print_blob();
    // done with while loop if:
    //   1. orange blob is centered in the camera,
    //   2. the turn increment is really small
    if(is_orange_visible_centered()) {
        printf("center: DONE: orange_centered\n");
        done=1;
    }
    if(fabs(turn_increment) < blob_locationthresh*0.1) {
        printf("center: DONE: turn_increment is small: %f < %f\n",
                fabs(turn_increment), blob_locationthresh*0.1);
        done=1;
    }

    //let other threads run
    rc_usleep(500);    // Needed for virtual lab version

}//endwhile
printf("----------END center_orange----------\n");
```

*Figure 2. center_orange()*

```
    // move forward until "orange" object is large:
    // do in increments of 10 cm
    printf("----------START forward-to-orange----------\n");

    forward(0.1);

    camera_process_next_frame();
    print_blob();

    // loop until we are close enough to bottle:
    //    1. adjust heading so we are pointing at the bottle
    //    2. go forward a little
    //    3. take a picture and process it.
    //    4. sleep
    while(!is_orange_visible_big()){
forward(0.1);
camera_process_next_frame();
recenter_orange();
        // code goes here....

        //let other threads run
        rc_usleep(500);    // Needed for virtual lab version
    }//endwhile
    printf("----------END forward-to-orange----------\n");
```

*Figure 3. forward_to_orange()*

3) Compare the reading of the front sonar with the measured distance to the bottle. Why are they different? What does this mean for determining the position of the bottle during a search-and-rescue operation?

The sonar reading was 16 cm while the distance from the bottle was 7cm. This difference could be due to factors like the sonar's cone-shaped FOV, the bottle's rounded circuit, the angle of the sensor mounted on the robot, or outside interference. In search-and-rescue operations, relying only on sonar for distance measurements would be unreliable in determining the bottle's location because our robot's measured distance had an error of over 100%.

4) Submit your values for measurements 3.1, 3.2, and 3.3. Since the camera has a field-of-view of 60 degrees and 640 pixels across, you can use this, as well as the camera distance to the object, to calculate the width of your object from the image saved in experiment 3. Use your favorite image viewer to obtain the width of the bottle in the image, in pixels. Then, use geometry to calculate the physical width of the bottle, in meters. Compare this with the measured width. Include a copy of **robot_out.jpg** as part of your lab write-up. Show your work.

Name: Joshua Lee 11-10-2025

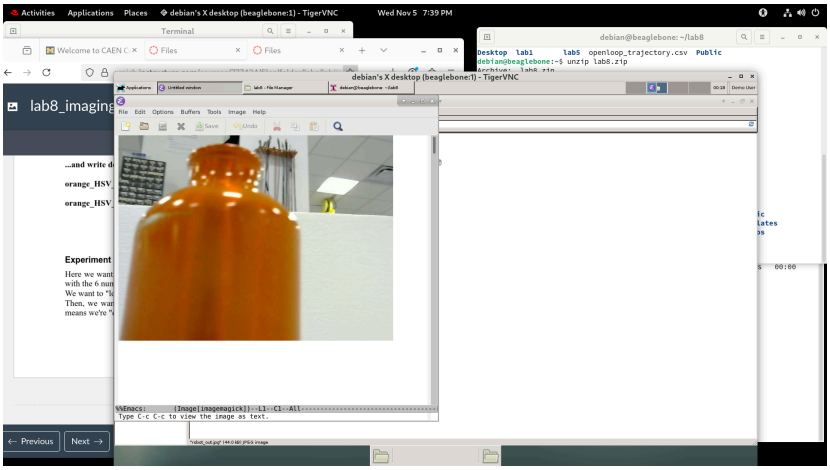

*Figure 5. Robot_out.jpg*

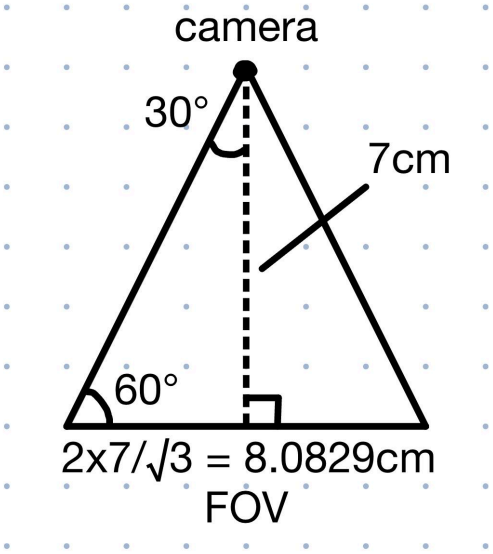


*Figure 6. Calculating the width of the camera's FOV*

3.1 Reading of the front sonar: **16cm**

3.2 Distance between camera lens and object: **7cm**

3.3 Width of water bottle: **7.25cm**

Based on *Figure 5* which displays the robot's camera output of the orange bottle, I found that the bottle was approximately 410 pixels wide.

Bottle width = 410 pixels
FOV width = 640 pixels

Name: Joshua Lee 11-10-2025

FOV angle = 60°
Distance from camera to object = 7 cm

Based on the values we measured in our lab and our knowledge of the camera specifications, we can calculate the robot's FOV to be 8.0829cm as shown in *Figure 6*. Since the width of the image is 8.0829 cm, multiplying it by the ratio 410px/640px gives us the width of the bottle in the image which is **5.17 cm**. The difference of 2.08 cm between the calculated and measured widths could be due to the bottle's angle in the image or inaccuracies in the pixel measurement.