HOME ASSIGNMENT FOR MACHINE VISION

Muhammad Nasir Sabir
230084

Esma Nur Ekmekci
230087

Lecturer: Professor Pekka Toivanen

Topic 1: Counting bottles in a crate

Abstract: We have been working on a solution to count the number of bottles in each crate, and although we tried various approaches, some were ineffective for certain photos. While we attempted specific solutions for these photos, we found that using too many corrections ultimately led us down a dead-end path. Therefore, we decided to try a different approach. Through several steps, we were finally able to count all of the bottles, including those with fallen pet cups, upside down bottles, and bottles with different lighting conditions. However, we were unable to count bottles that had fallen sideways, as our solution was based on detecting circles.

First Approach

Our main approach involved utilizing mathematical morphology techniques such as opening and closing. To begin, we binarized our images using a threshold value that we determined during the image reading process. We performed this process twice: once to detect even the dimly lit areas using a lower threshold value *[figure 1.1]*, and the second time to detect the circles within the bottles using a higher threshold value that was just before losing circle connectivity *[figure 1.2]*.

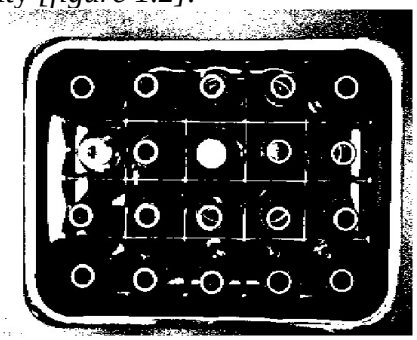


figure 1.1

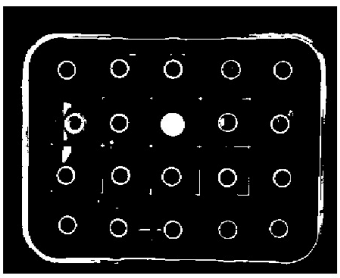


figure 1.2

After applying an opening method to our lower thresholded image, we retained only the brightest areas by using structural elements such as "line" and "disk" *[figure 2.1]*.The main purpose of this step was to subtract the brightest areas, such as the edges of the crate, from our second (higher) thresholded image *[figure 2.2]*.

Ultimately, our goal was to preserve the connectivity of circles in order to fill any holes, while simultaneously disrupting the connectivity of larger, bright areas to prevent them from being filled in. So far, this approach has been effective in producing satisfactory results.

figure 2.1

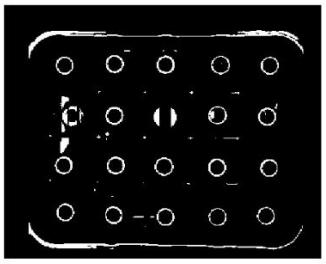


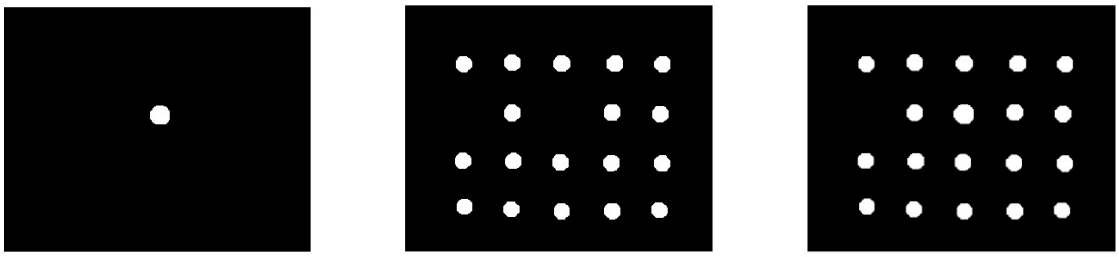
figure 2.2

However, we encountered a challenge with images of bottles that had shiny cap rings. Our previous step removed these areas as well. To address this issue, we once again thresholded the image third time *[figure 3.1]*. This time ensuring that the brightest bottle cap areas we had removed in the previous step were preserved. We then applied the opening method to this thresholded image, resulting in circles of the brightest bottle caps *[figure 3.2]*. Next, we combined this output with the previous output we had obtained. With these solutions, we were able to detect the bottles in almost all cases. However, we still encountered difficulties with hard-to-read images such as upside-down bottles and fallen pet glasses, and were unable to accurately count these instances.

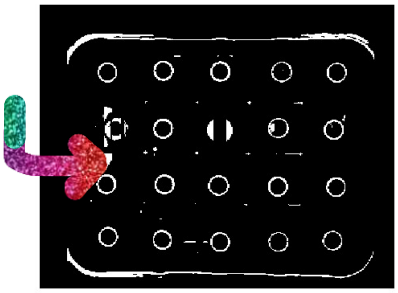
figure 3.1

figure 3.2

After experiencing these failures, we considered that using methods based on Hough transform could be more effective in detecting circle shapes, even under hard-to-read circumstances.



Combining two results. But still there is one hole missing.



The reason for this is that the missing bottle cap in this particular image is not bright enough to be detected in the third thresholding step, and also it is much brighter than our second threshold. After we applied opening and subtraction to remove bright areas, that step caused it to lose its connectivity and affected the missing cap area to become disconnected and prevented becoming a fully circle.

CODE:

```
seLine=strel('line',40,90);
seDisk=strel('disk',2);
seDisk2=strel('disk',6);
seDisk3=strel('disk',15);
seDisk4=strel('disk',5)

%Image reading
A=imread('images/bottle_crate_24.png');

%Tresholding for bright full white bottle caps
kapakliA=im2bw(A,.99);
imshow(kapakliA);

%Image filling by holes for bright bottle caps, If any without full circle
diskKA = imfill(kapakliA,"holes");
imshow(diskKA);

%Closing for improve connectivity
KAclosed = imclose(diskKA,seDisk);
imshow(KAclosed);
%Opening for just remain the circles
KAopened = imopen(diskKA,seDisk3);
imshow(KAopened);



%First treshold with low treshold
backgroundofA=im2bw(A,0.20);
imshow(backgroundofA);

%Secon treshold with higher treshold
frontofA=im2bw(A,0.38);
imshow(frontofA);

%Opening with 'line' structring element to leave just the brightest areas
openedBA= imopen(backgroundofA,seLine);
imshow(openedBA);

%Subtracting the left bright areas after opening from second(higher) treshold
subtractedA=imsubtract(frontofA,openedBA);
```

```
imshow(subtractedA);

%Filling circles
diskA = imfill(subtractedA,"holes");
imshow(diskA);

%Remove the elements that is not a full circle
lastA= imopen(diskA,seDisk3);
imshow(lastA);

%This part is for fixing matrix pixel values after subtracting.
%Converting -1 values to 0.
for ii=1:size(lastA,1)
   for jj=1:size(lastA,2)
      % get pixel value
      pixel=lastA(ii,jj);
       % check pixel value and assign new value
       if pixel<0
          lastA(ii,jj)=0;
       end
       % save new pixel value in thresholded image
    end
 end

%Combining the results between bright full bottle cans and normal circles
sumofA = or(lastA,KAopened)
imshow(sumofA)
```

Second Approach

After the first approach, we thought that other methods may lead to more effective results for given problem. So we mainly focused on "*Hough Transform*" since we were trying to find circles of bottles to count them. Before applying hough transform, we consider making the image more convenient for our algorithm to work. As a result, these are the steps of our second approach:

Threshold: To separate objects or regions in an image based on their intensity values, we used threshold. It is converted to a binary image based on a threshold value.

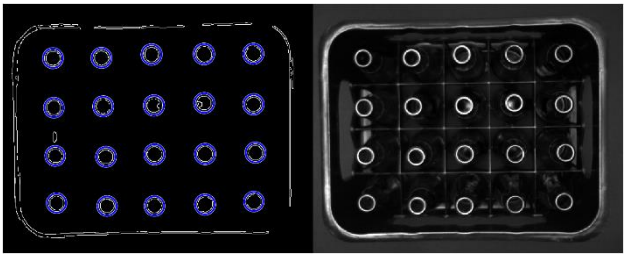
Opening: By applying opening, we remove noise or small objects that may not be relevant to the analysis.

Edge detection: We tried several edge detections including *'Prewitt', 'LoG'* and *'Canny'*. So, we decided on Canny method since this problem requires accurate edge detection and localization. This method minimized false detections.

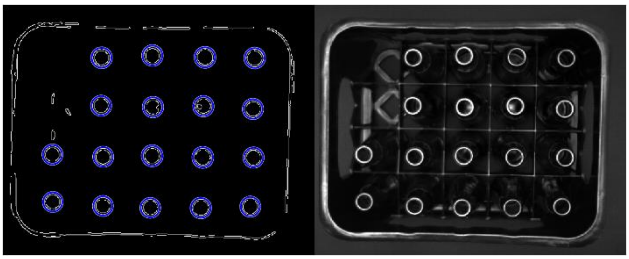
Hough Transform: This method is designed to identify geometric shapes in an image. For our case, we need to define circles to count bottles from a bird's-eye view. So, it means that we are unable to count slanted bottles. Radii are important, with minimum radius we find flat standing bottles and with maximum radius we find flipped bottles. Not to take cups with a large radius, we observed by giving different numbers to maximum radius. Thus, we find a number to take flipped bottles but not the cups.

We managed to find all bottles except the slanted bottles.

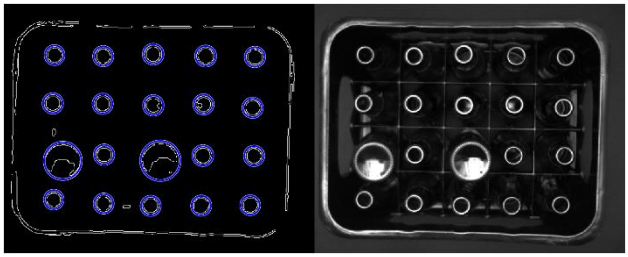
Results From Second Approach



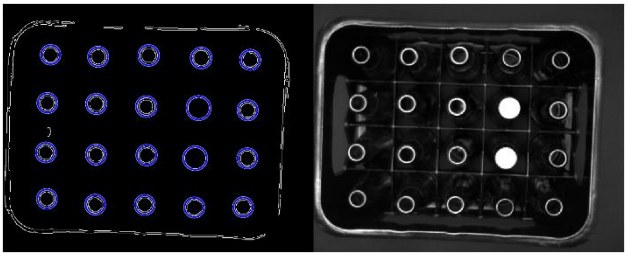
bottle_crate_01



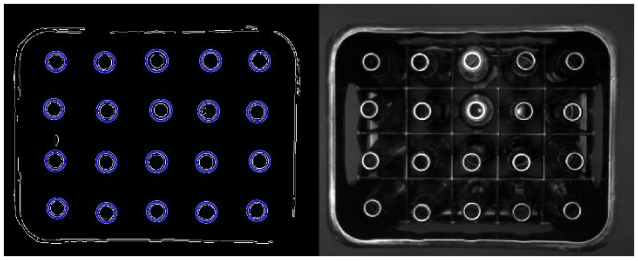
bottle_crate_02



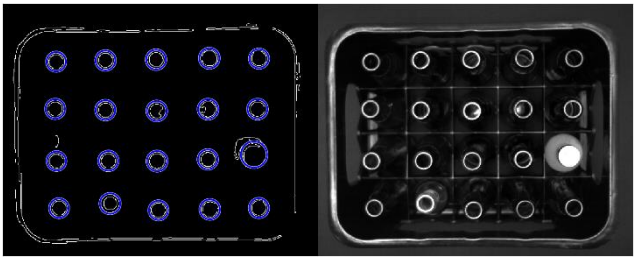
bottle_crate_03



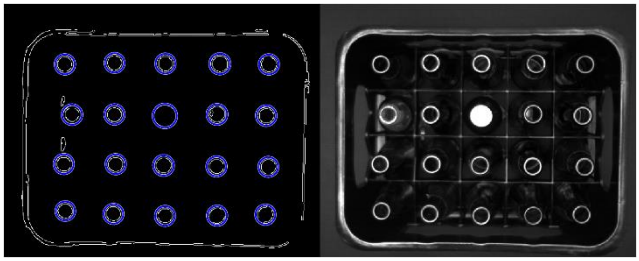
bottle_crate_04



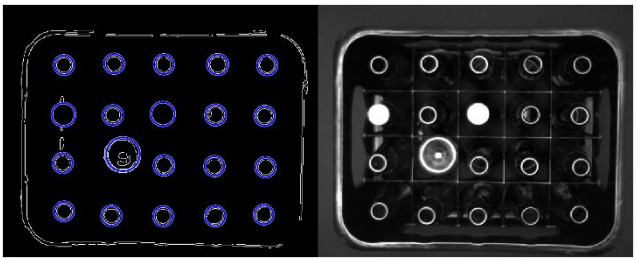
bottle_crate_05



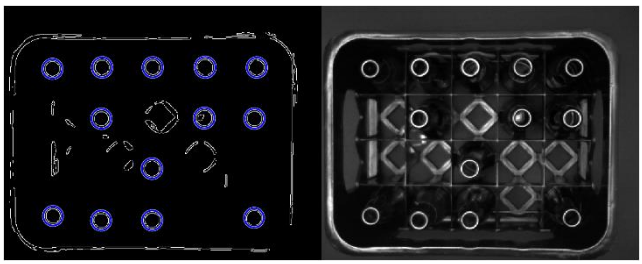
bottle_crate_06



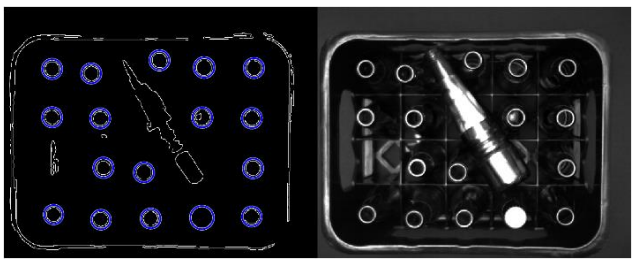
bottle_crate_07



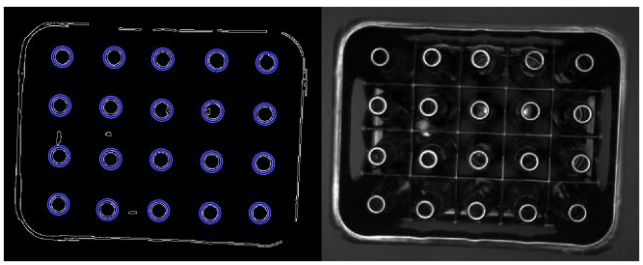
bottle_crate_08



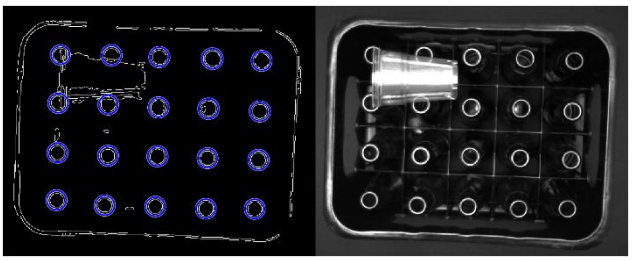
bottle_crate_09



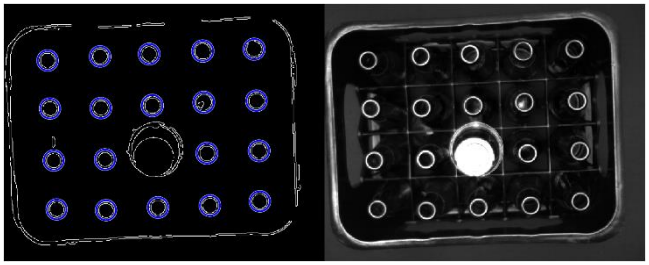
bottle_crate_10



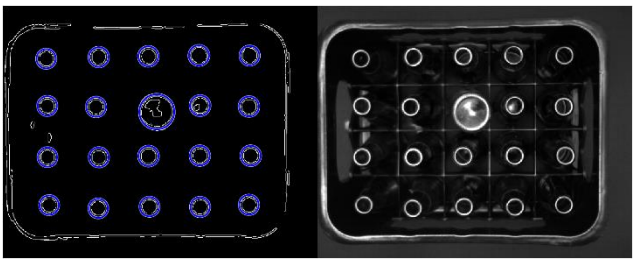
bottle_crate_11



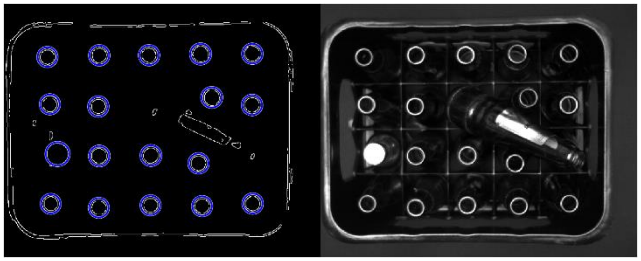
bottle_crate_12



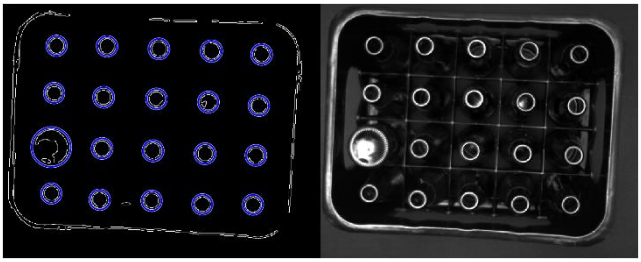
bottle_crate_13



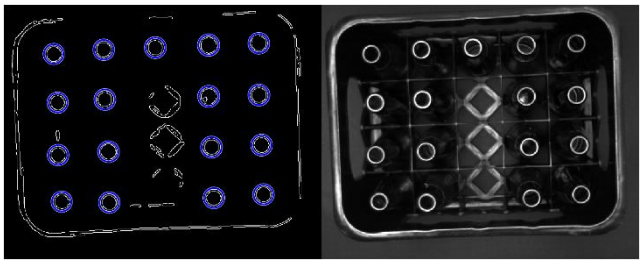
bottle_crate_14



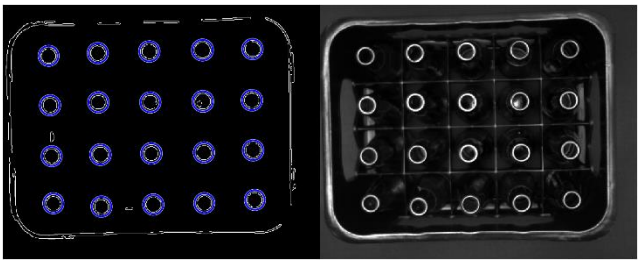
bottle_crate_15



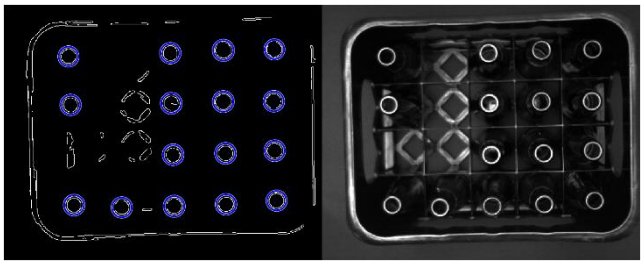
bottle_crate_16



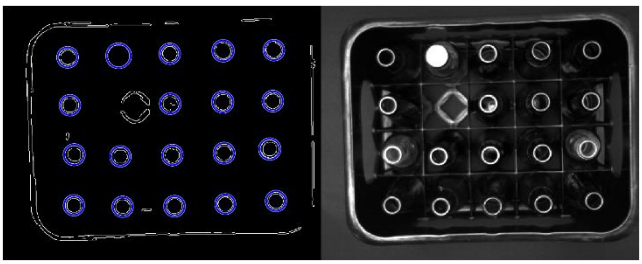
bottle_crate_17



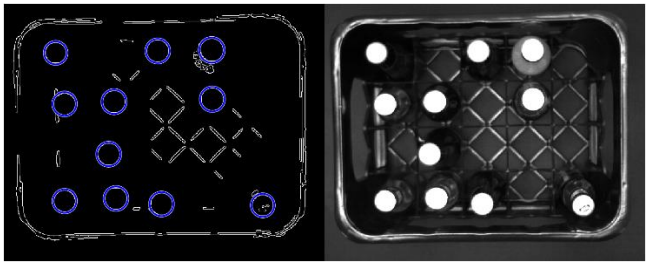
bottle_crate_18



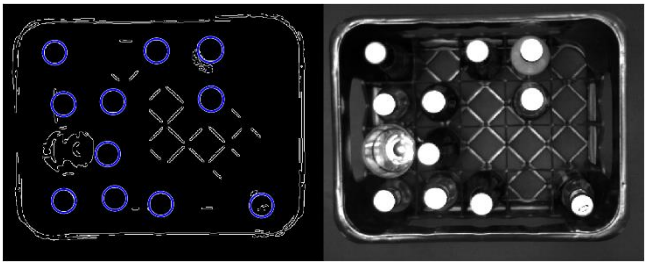
bottle_crate_19



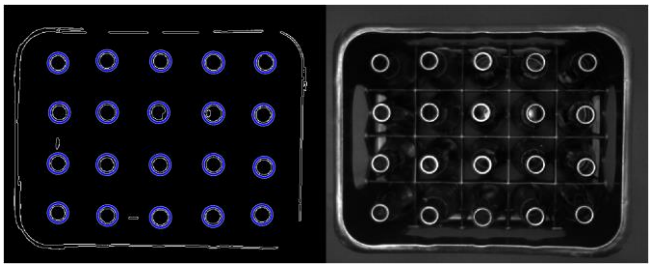
bottle_crate_20



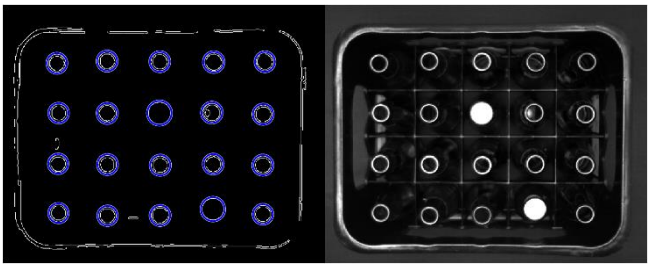
bottle_crate_21



bottle_crate_22



bottle_crate_23



bottle_crate_24

CODE:

```
img=imread('images/bottle_crate_24.png'); %read image

thImg = imbinarize(img,.524); %convert image to a binary image based on threshold
bwImg = bwareaopen(thImg,50); %remove small objects

cannyImg = edge(bwImg,'canny',[]); % Canny edge detection

min_rad = 17; %for normal bottles
max_rad = 42; %for flipped bottles
[centers, radii, metric] = imfindcircles(cannyImg, [min_rad, max_rad]); %search for circles with given
radii using circular Hough transform
figure
imshowpair(cannyImg,img,'montage'); %show given two image at the same time

viscircles(centers, radii, 'EdgeColor', 'b'); %show detected circles
```