Assignment 1

Basic Image Processing Algorithms
Fall 2022

General rules

This is the first Assignment. There will be three more Assignments during the semester.

Point value: **20 points**, which is 20% of the total assignment points.

Deadline: **October 3, 2022 23:59:59** (late submission until Oct. 5)

This is a not-guided exercise. The description of this assignment is general and does not focus on the details as in case of the Lab exercises.

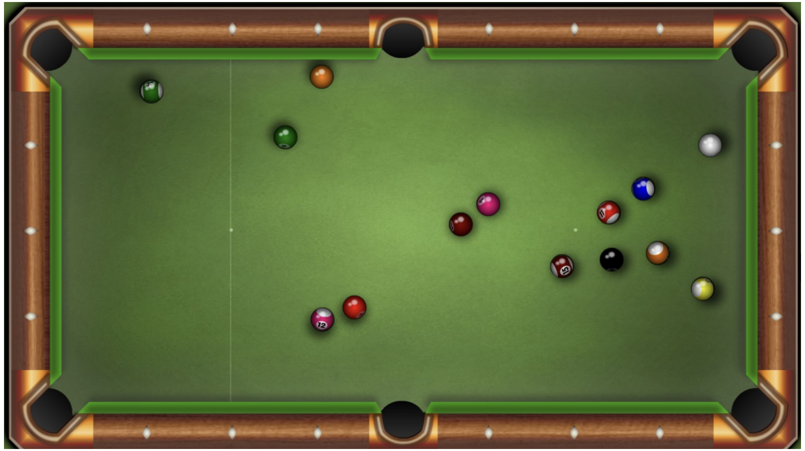
The main task is to provide a good, reasonable solution. You may code “freely” (only minimal restrictions on file names and outputs are given).

Problem formulation

You have to write a custom detector and identifier program that can work on different images with very good confidence.

Input: an image showing an 8-ball pool table with pool balls

Output: figures and console output (presence and position of each ball)



Tasks to do

Write a script that…

1. loads the sample image into a variable,
2. applies histogram operations (e.g. stretching) to enhance quality
3. creates a binary mask using the image of an empty table and thresholding
4. applies a ball-detecting conv. kernel to find the round objects
5. crops the round objects from the image one-by-one
6. for each round object it gives a 2D position as a coordinate in cm
7. if the object is a ball, quantize its value based on color (histogram)
8. for each possible ball number print a list in which the presence and position info is listed (i.e. is the ball with value X visible and if yes, where?)

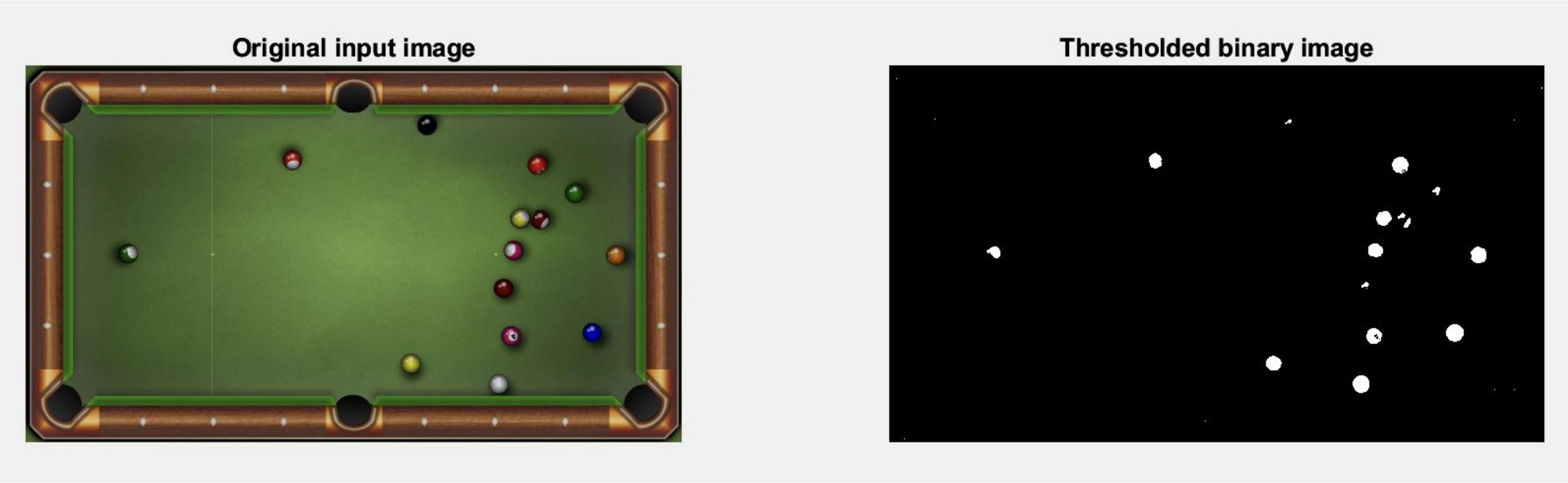
Key results to be presented:

You may code freely, as there are no restrictions on what functions, variable names and processing flow to use.

However, please create the following outputs:

1. **Figure 1** should show the original and binary image side-by-side
2. **Figure 2** should show the result of the convolution
3. **Figure 3** should show the cropped round objects, and for each subfigure the title says the 2D position in cm
4. **Figure 4** should show a similar image to Fig. 3 but after value-detection, showing the determined value (number) of the balls
5. The answer (report of coords.) should be displayed on the **console**

Figure 1



Original input image

Thresholded binary image

Figure 2

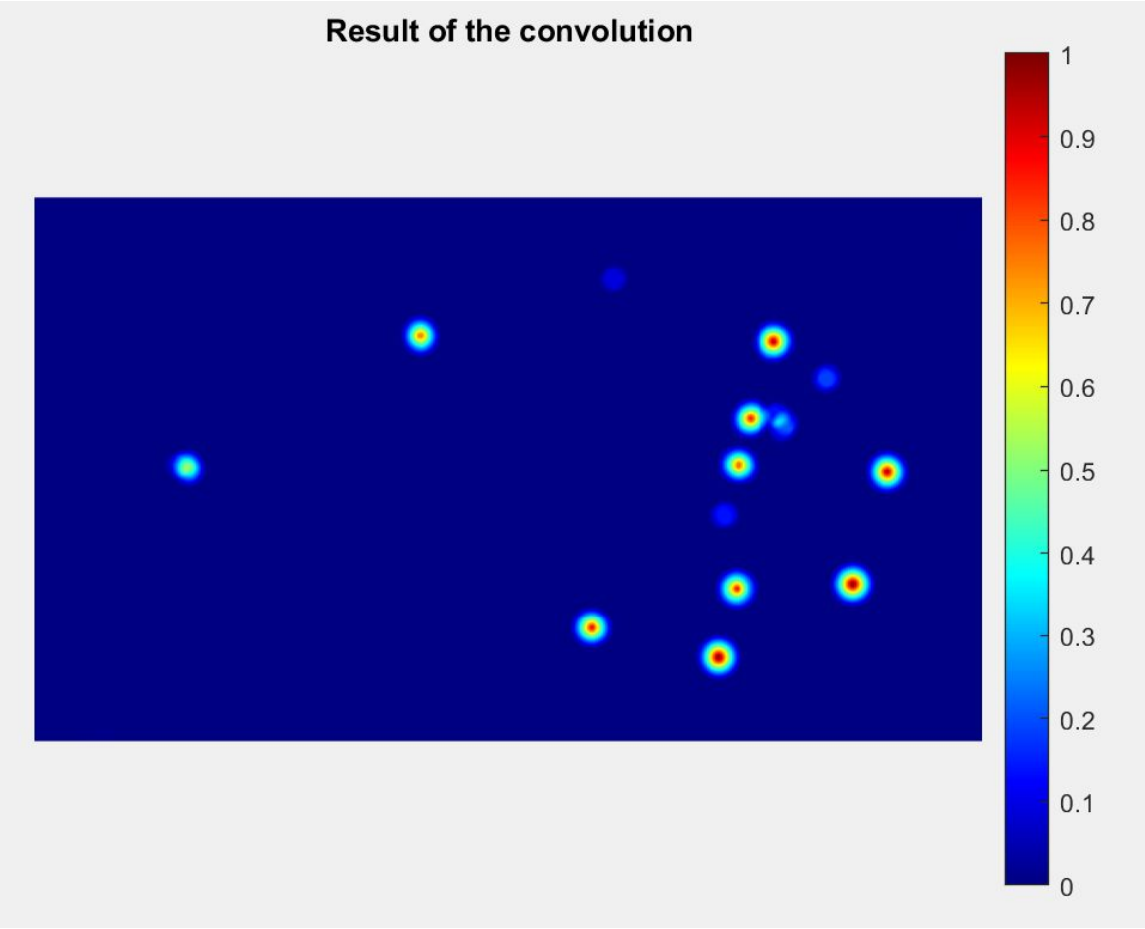


Figure 3

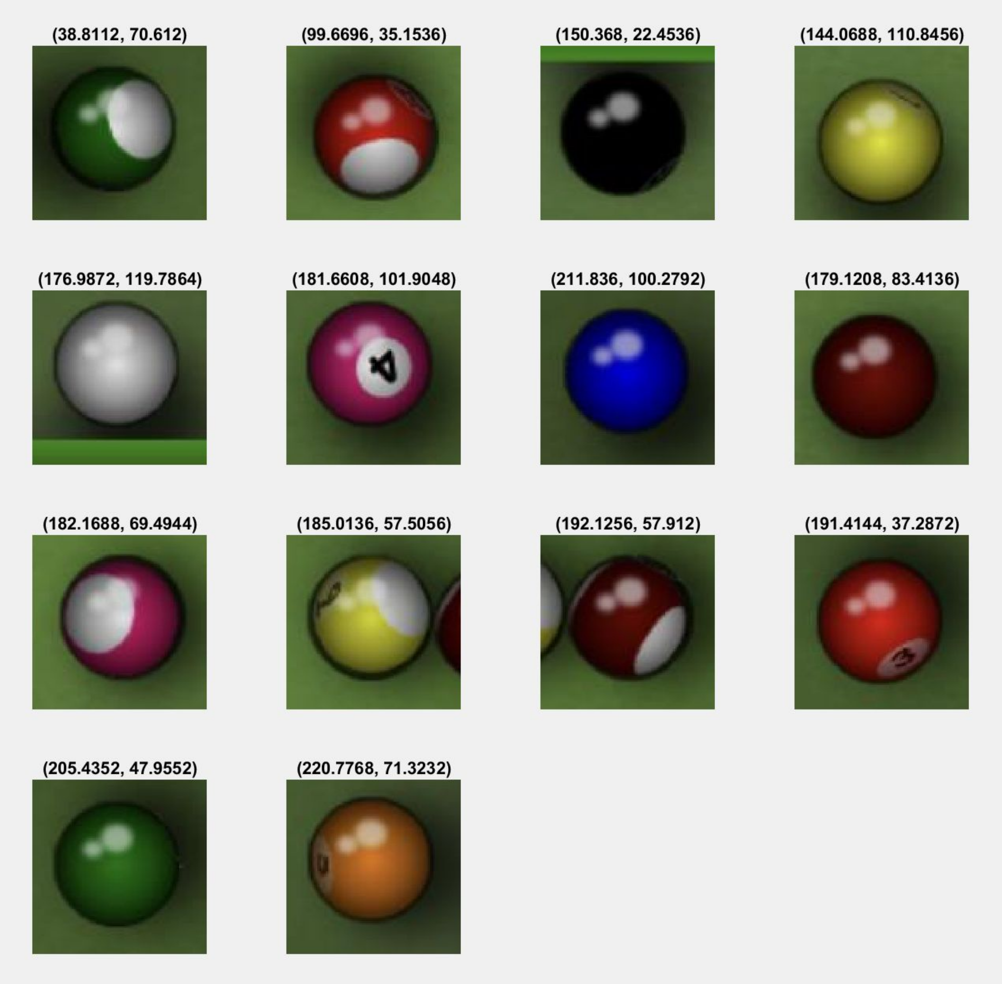
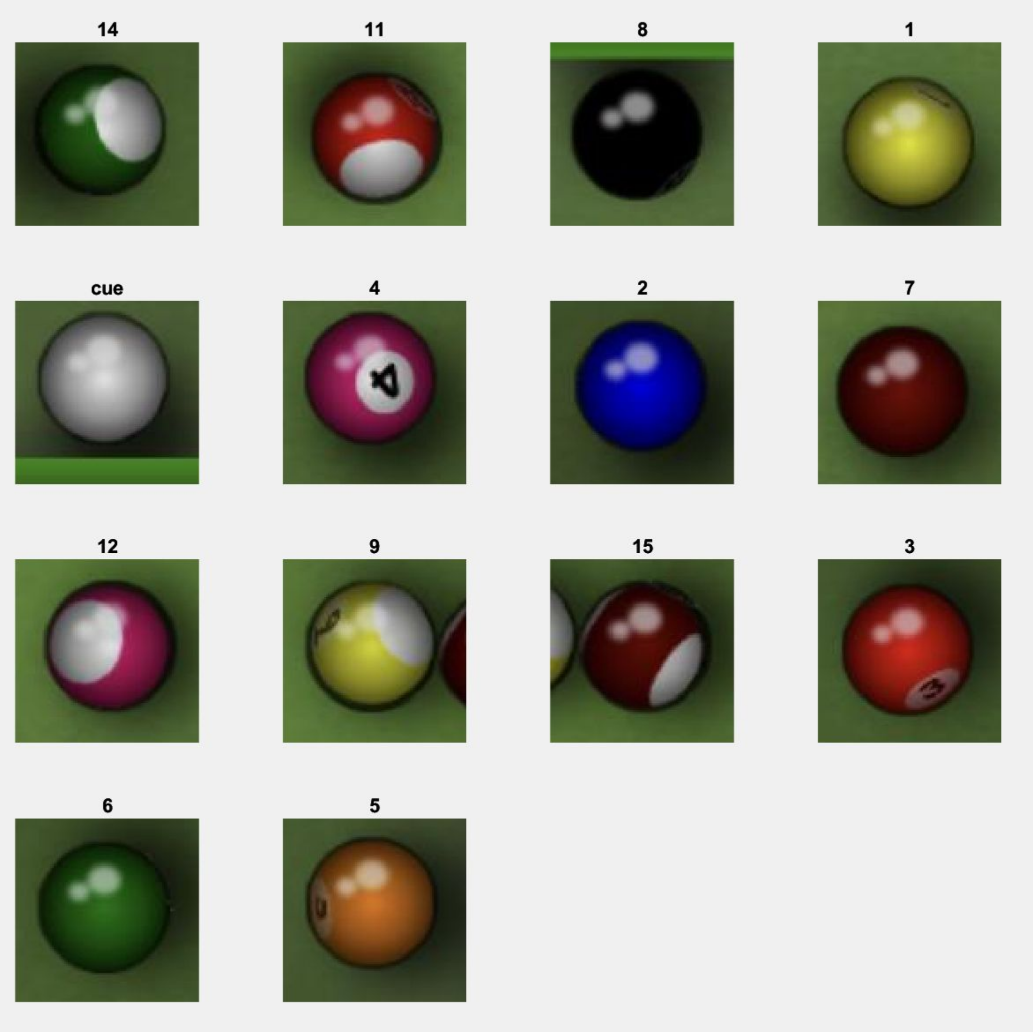
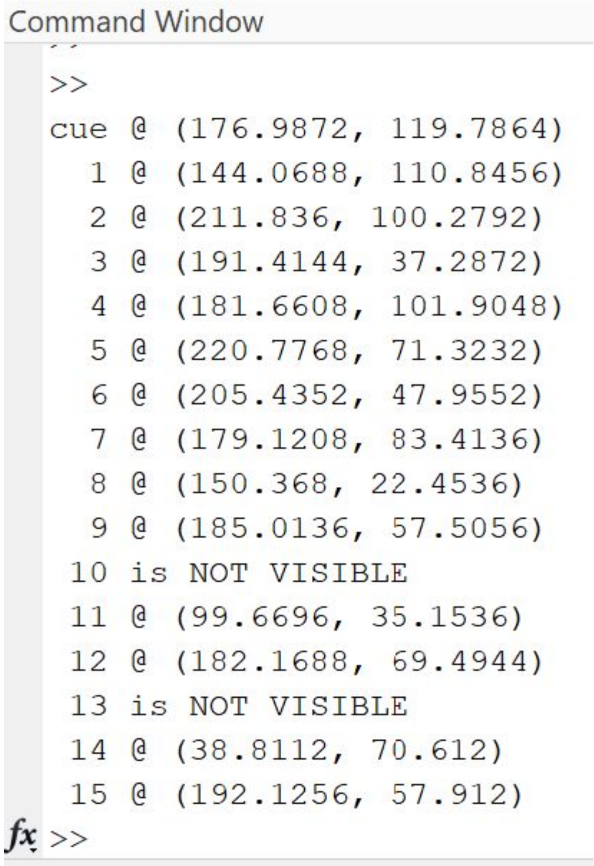

(38.8112, 70.612)
(99.6696, 35.1536)
(150.368, 22.4536)
(144.0688, 110.8456)
(176.9872, 119.7864)
(181.6608, 101.9048)
(211.836, 100.2792)
(179.1208, 83.4136)
(182.1688, 69.4944)
(185.0136, 57.5056)
(192.1256, 57.912)
(191.4144, 37.2872)
(205.4352, 47.9552)
(220.7768, 71.3232)

Figure 4



The console output:



```
>>
cue @ (176.9872, 119.7864)
  1 @ (144.0688, 110.8456)
  2 @ (211.836, 100.2792)
  3 @ (191.4144, 37.2872)
  4 @ (181.6608, 101.9048)
  5 @ (220.7768, 71.3232)
  6 @ (205.4352, 47.9552)
  7 @ (179.1208, 83.4136)
  8 @ (150.368, 22.4536)
  9 @ (185.0136, 57.5056)
 10 is NOT VISIBLE
 11 @ (99.6696, 35.1536)
 12 @ (182.1688, 69.4944)
 13 is NOT VISIBLE
 14 @ (38.8112, 70.612)
 15 @ (192.1256, 57.912)
fx >>
```

There is no code package for this assignment.

All scripts and functions must be written entirely by you.

Download the images to be processed from here:
https://beta.dev.itk.ppke.hu/bipa/assignment_01

Submission & hints

You should create a script named `a01_NEPTUN.m` where the NEPTUN part is your Neptun ID. This has to be the main script; running that must be able to solve the problem.

You are allowed to create other files (e.g. additional functions) too, if necessary.

Please submit ALL files (including the image as well) in a compressed **ZIP** file via the Moodle system.

Check the upcoming slides for hints!

Hint 1

Stretch the histogram of the image to the full dynamic range. Use the image ‘empty.jpg’ and thresholding on separate color channels to get a decent mask. (E.g. you may subtract the ‘empty’ image from the one you are processing to get rid of the background)

A mask with large white areas in non-ball objects will cause problems.

Aim for a mask that is white in the ball regions.

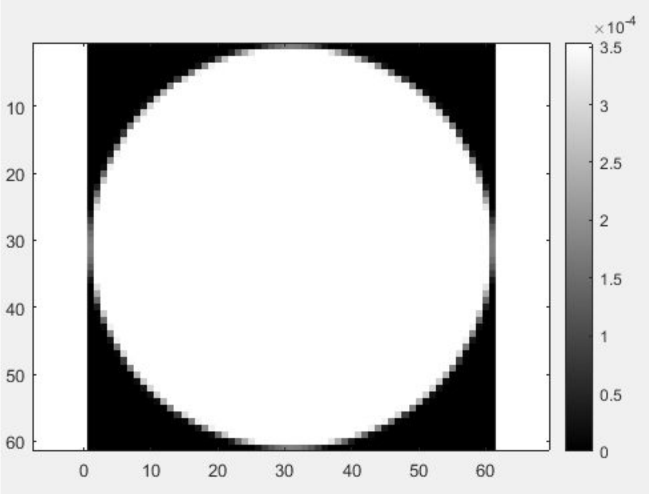
If the mask is “noisy” (small white dots) it’s not a problem, the convolution will deal with it.

Hint 2

The “ball-detection kernel” should be a round kernel with a diameter comparable to the diameter of the balls.

Something like this will do:

(The size of a good kernel might be different in your case.)



Use the MATLAB’s built-in kernel generator function to obtain such matrix.

Hint 3

Cropping the objects may sound hard but it isn’t.

Define a window size that is large enough to contain the whole ball.

Find the location of the maximum in the convolved image, and crop the window-sized area around the same spot in the original, color image.

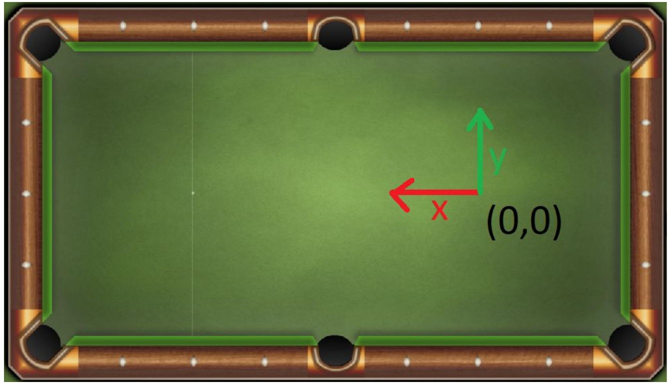
Modify the result of the convolution; replace every value inside the window with 0. This will destroy the now-found maximum and the next iteration will find the next max value, possibly a new ball for which the kernel shows significant activation.

Stop the loop when the value of the maximum is not significant anymore.

Hint 4

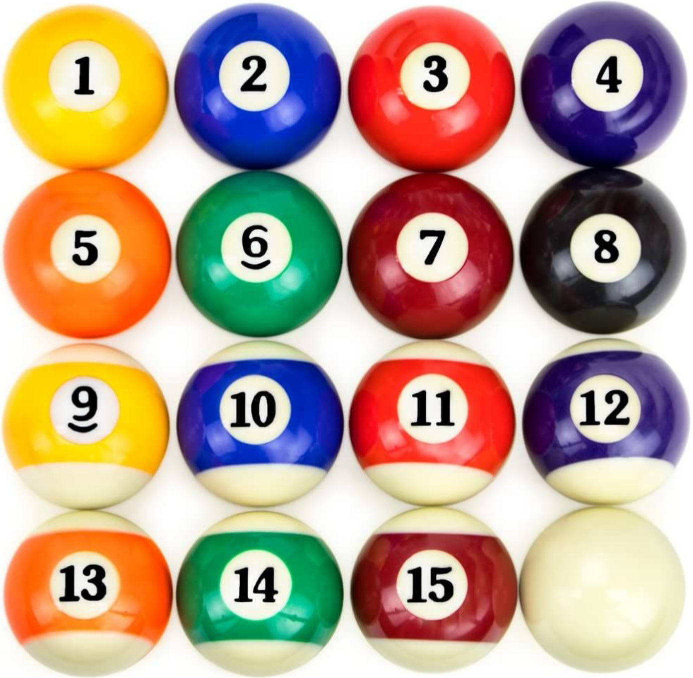
To get the coordinates of the balls in centimeter (cm) use the following idea: We know that the standard pool table has a size of 44” × 88”. By measuring the size of the image, this gives a result like 1100 px = 44”. Convert the 44 inches to centimeters and you get a formula for how many cm does a pixel mean in the image.

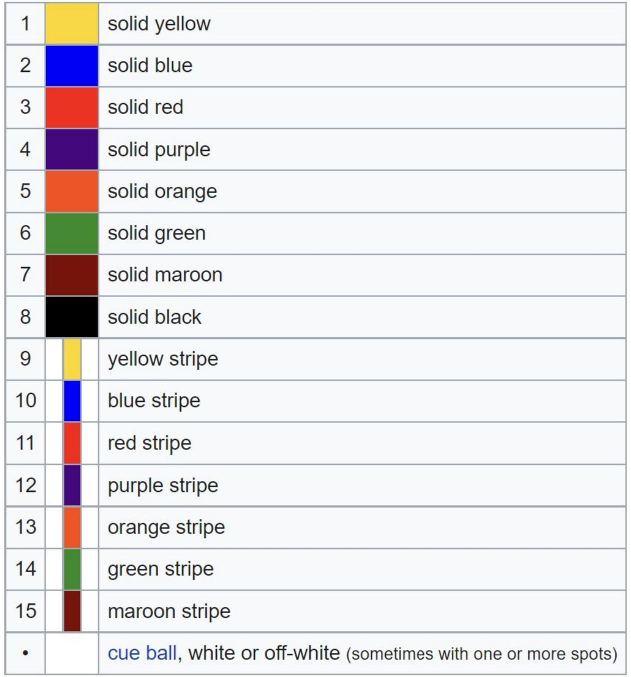
The cm coordinate system is shown here. The origin is at the ‘foot spot’.



Hint 5

The histogram of the cropped color image can be useful to decide the value of the ball. The 8-ball pool game uses a standard color scheme:





1	🟨	solid yellow
2	🟦	solid blue
3	🟥	solid red
4	🟪	solid purple
5	🟧	solid orange
6	🟩	solid green
7	🟫	solid maroon
8	⬛	solid black
9	🟨	yellow stripe
10	🟦	blue stripe
11	🟥	red stripe
12	🟪	purple stripe
13	🟧	orange stripe
14	🟩	green stripe
15	🟫	maroon stripe
•		cue ball, white or off-white (sometimes with one or more spots)

Hint 6

You have to build a so-called “look-up-table” in which you store the possible ball values (cue and from 1 to 15).

Create it manually, the initial values for each ball should be “not visible”.

During processing the image, if you found a ball, associate the found coordinates with that value.

Finally, you have to print this table to the console.

Grading

The final score of this assignment is the sum of the following points:

The script filename is correct, it’s a script, the image is loaded, no errors	2 points
Figure 1 exists, similar to the sample in this document, mask is good	2 points
Convolution kernel exists, size and kernel values are OK	1 point
Figure 2 exists, similar to the sample in this document, result is good	2 points
Object cropping is done in a loop with a window, at least 70% found	3 points
Coordinate detection, unit conversions working correctly	2 points
Figure 3 exists, similar to the sample in this document	2 points
Histogram or color based value detection exists and looks good	2 points
Figure 4 exists, similar to the sample in this document	1 point
Console output exists, the values are OK according to Fig. 3 & 4	1 point
Code quality (readability, understandability, good comments and structure)	2 points
TOTAL:	**20 points**

Contact

If you have any further questions regarding this assignment, contact

Márton Bese NASZLADY

via **Teams** (in private chat) or write an email to

naszlady@itk.ppke.hu

THE END