Introduction to Visual Computing

Assignment# 13 – [**OPTIONAL ASSIGNMENT**]

Blob-based detection of obstacles

May 22, 2018

Description

This last assignment aims at completing the game experience by detecting the 3D printed objects present on the Lego board, and initializing the game obstacles with them.

Objectives

Detect the 3D printed objects using a blob-detection algorithm, and use them to initialize the (virtual) obstacles of the game.

Specific Challenges

To filter the blob according to color and position within the quad and to map the coordinates to your board.

Part I

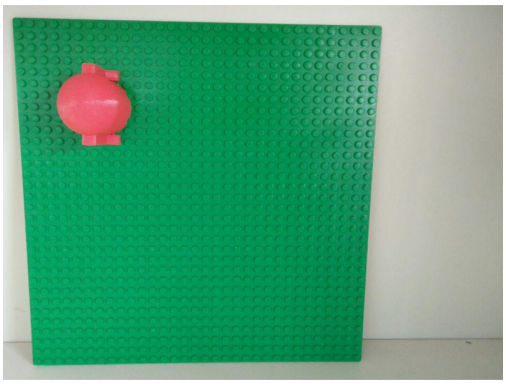
Blob detection

In order to detect obstacles on your board, you have to use a pipeline similar to the one to detect the green blob of the board. First, color threshold the image according to the color of the obstacle. Then use the blob detection class to find the blobs. Few steps need to be added for a successful integration.

Step 1 – Compute the centroid of blobs

Create a function that computers the center of each blobs by simply averaging the x and y coordinates of all the points in the blob.

You can test you estimation of the location of the blob using the following image. On this image the object is placed at $6cmX6cm$ from the top left corner of the boar. And the center of the blob is about at the pixel $[140, 160]$.



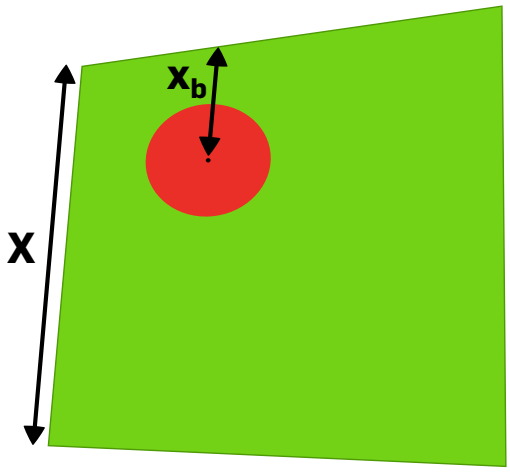
Step 2 – Keep only blobs inside the green board

Use the following code to check that a point is inside the valid quad.

```
public boolean isInsideQuad(int x, int y, ArrayList<PVector> quad) {
        PVector A= quad.get(0);
        PVector B= quad.get(1);
        PVector D= quad.get(3);

        PVector AB=PVector.sub(A, B);
        PVector AD=PVector.sub(A, D);
        PVector AP=PVector.sub(A, new PVector(x, y));

        float productABAP=AB.x * AP.x+AB.y * AP.y;
```

```
        float productABAB=AB.x * AB.x+AB.y * AB.y;
        float productADAP=AD.x * AP.x+AD.y * AP.y;
        float productADAD=AD.x * AD.x+AD.y * AD.y;

        if ( (productABAP > 0) && (productABAP < productABAB) &&
        (productADAP>0) && (productADAP<productADAD) )
        return true;

        return false;
}
```

Step 3 – Estimate the location of obstacles on the virtual game board



Estimate the location of the center of the blob in the board coordinates with a simple proportion:

$$\frac{X}{s_{board}} = \frac{x_b}{x_{obstacle}} \Rightarrow x_{obstacle} = \frac{x_b}{X} s_{board}$$

with X the length of the board's side in the image, s_{board} the size of the board in the virtual game, x_b the x coordinate of the blob relative to the board edge (as an approximation, assume that the board is facing perfectly straight the webcam) and $x_{obstacle}$ the x coordinate of the blob center in the virtual game.

Part II

Integration with the game

Integrate the blob detection into your game: when the user presses `Shift` (to manually add obstacles with the mouse), try to detect the obstacles from the webcam stream, and automatically add them as well. Stop the blob detection once the user leaves the "Board configuration" mode (ie, when `Shift` is released.)