# Computer Vision Project 1

**Olteanu Fabian Cristian**

FMI, AI Master, Year 1

---

## 1. Introduction

In the following section I will explain the approach which was used in order to achieve a relatively modest score of 1.935 on the regular tasks (from running the `evaluate_submission.py` script).

## 2. Description of the Approach

In broad terms, the following steps were implemented to achieve the result:

1. The second image from the `board+dominoes` folder was used as a template whose features were extracted for SIFT matching with all other images used for training. I chose this picture because the perspective of the camera is the closest possible to being parallel to the table as opposed to the other pictures.
2. After the SIFT matching, the train pictures are cropped using some hand-picked values, rendering the board region of interest (SIFT matching changes the perspectives of the pictures to almost the same view amongst all of them, so cropping them using hard-coded values is fine in this case, since the regions of the board are almost the same across all images).
3. After the cropping, the pictures' perspectives are further warped in an attempt to perfectly align the boards' horizontal and vertical lines as perfectly parallel to eachother by calling the OpenCV `cv2.getPerspectiveTransform` method using as parameters some hand picked pixel values representing the region of interest containing the warped board (taken from the unwarped `align_train_1_01.jpg` image produced from an earlier version of the script) and the positions of the corner pixels of the same image. The output gives an almost perfectly aligned image in all cases, like in figure 1.
4. Template matching is done on all train images using `align_train_1_01.jpg` as the template. The output is a grayscale image with patches ready for classification. This results in pictures being tiled like in figure 2.
5. The `score_game.ipynb` script begins to first classify each patch based on whether or not it considers it to be a domino tile. This is achieved by comparing the whites and blacks from the binary image of each patch (from a hand-picked threshold) with the minimum and maximum whites and blacks from some cropped template domino tiles (from `boards+dominoes/07.jpg`).
6. After the classifier decided whether the patch is a domino tile or not, sometimes it encounters false-positives, like in the case of the patches in the middle of the board. In this case, they are be avoided when the classifier computes the number of the dots in the tile. This is done via the OpenCV implementation of the Hough Circle Transform, which uses some hardcoded `minRadius` and `maxRadius` parameters to determine only the small circles from dominoes and exclude false-positives.
7. When the classification is done, the positions and scores are calculated and stored in text files in an output specified in the script.

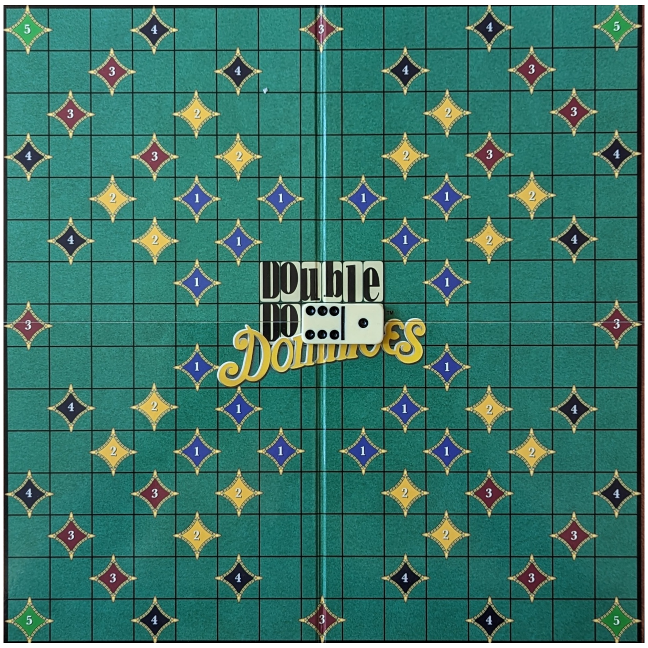

**Figure 1.** Aligned Picture of Move One from Game One

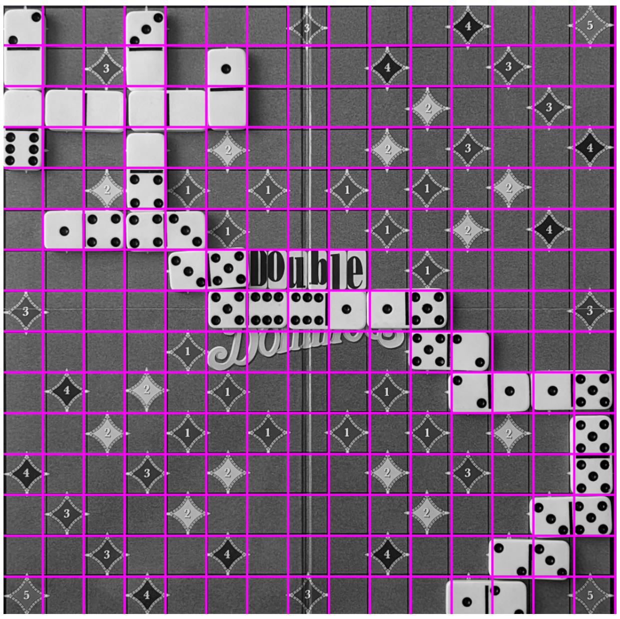

**Figure 2.** Aligned Tiled Picture of Move 20 from Game One