TASK 1

Signal Processing and Machine Learning

Image Processing

ArUco Marker

Introduction

An ArUco marker is a NxN grid or Bit size, that is black and white in color, where N=4 to 7.
ArUco markers are based on Hamming code.
Consider for example an ArUco marker of 5x5 grid or Bit size as shown in Figure 1 below

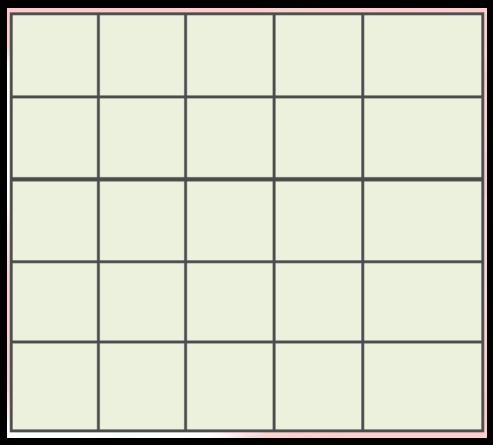

Figure 1: 5x5 grid of an ArUco marker

In this grid, the first, third and fifth columns represent parity bits. The second and fourth columns
represent the data bits. This is depicted by Red and Blue arrows in alternate columns in Figure 2
below; where Red arrow represents parity bit columns and Blue arrow represents data bit
columns.

Thus, from 2 data bit columns there are a total of 10 data bits. So, the maximum number of

markers that can be encoded are: 2^10 = 1024. Thus, a 5x5 ArUco marker can have a maximum
of 1000 combinations or IDs.

Note: There are two data bits in each row.

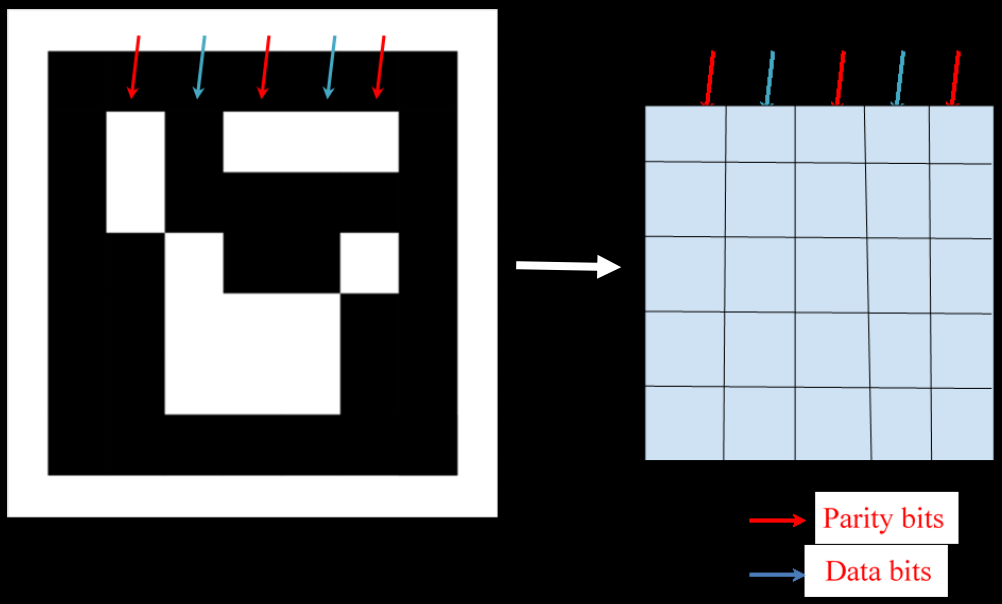

Figure 2: 5x5 ArUco marker

Encoding

Suppose we choose the example ID of 650 i.e. ArUco ID=650; its binary representation is 1010001010 i.e. 10 Bits.
Each row of the grid is encoded separately using a slightly modified Hamming code for each
parity bit i.e.
1. The first parity Bit is calculated using even parity
2. The second parity Bit is calculated using odd parity
and
3. The third parity Bit is calculated using even parity

The ID 650 is generated according to the above Hamming code and we get the following
encoded values, as shown in the Bit table - Table 1 below:

Table 1: ArUco ID = 650 Bit table (parity and data bit values)

Data 2	*Parity 3*	*Data 1*	*Parity 2*	*Parity 1*
0	*0*	*1*	*0*	*1*
0	*0*	*1*	*0*	*1*
0	*0*	*0*	*1*	*0*
0	*0*	*1*	*0*	*1*
0	*0*	*1*	*0*	*1*

By rearranging the bits in each row, the following data rows as shown in Figure 3 are obtained:

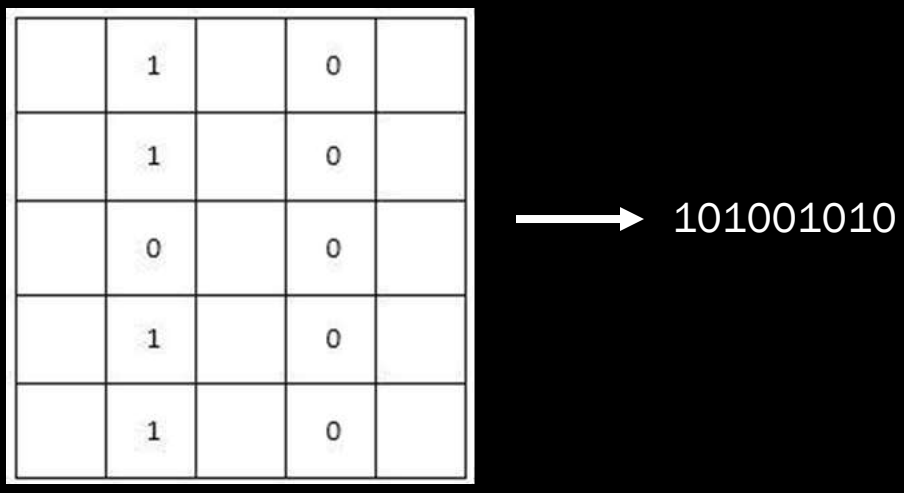

Figure 3: Data Rows

Rearranging the parity bits as well, Bits for ArUco ID = 650 look as shown in Figure 4:

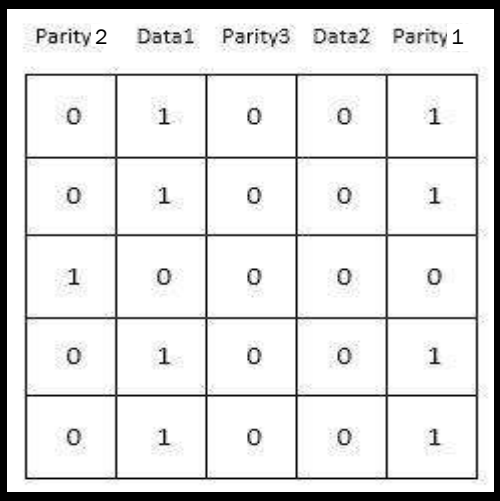

Figure 4: Bits for ArUco ID = 650 (0b1010001010)

Cells having value 0 are represented by black coloured pixels; while the cells having value 1 are represented by white coloured pixels as shown in Figure 5 below:

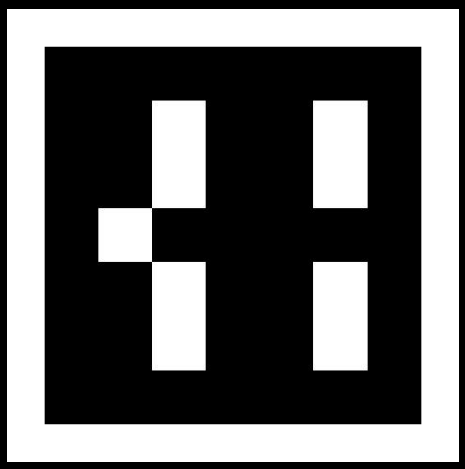

Figure 5: ArUco marker for ID=650

The encoded ArUco marker image is padded with a single layer of black cells. This layer will
be removed during decoding.

Decoding

After understanding the above section, decoding is an extremely simple process. The following
steps are to be followed while decoding a perfect, computer-generated image of an ArUco
marker.

Step 1: Extract the ArUco from the image.
Step 2: Remove the extra padding.
Step 3: Divide the resulting image into a NxN grids and check the color in each cell of the alternate
columns in a top to bottom manner, starting from the second column.
Step 4: If the color is white, write 1; else, write it 0.
Step 5: The resulting number will be in binary. Convert it into decimal.

Your Task:

The procedure given above is followed in the Original ArUco Dictionary, which has only 5x5 markers in it. However, with the advancement in technologies, several other dictionaries have been added to the ArUco library such as DICT_4x4_[Num], DICT_5x5_[Num], DICT_6x6_[Num] etc. where underlying principle is still hamming code, but along with it, certain other mathematical computations are applied such as Probability distribution. Implementing those is beyond the scope of this task.

Hence, in this task you have to implement only *'Original ArUco Dictionary'*. The test images given to you, doesn't belong to Original ArUco dictionary, hence discard those images. For test images either you can use the ArUco library to generate one or you can visit http://chev.me/ArUcogen/ . There select the required ID and dictionary and you can generate the ArUco marker by yourself.

The task is having two parts:

1. *Detection of Marker:*
   You have to write a script which can detect all ArUco markers (falling under 'Original ArUco' dictionary) present in a given image. We will test your code with several test cases, so don't restrict the code and hence try to make your code as much as generic as possible.

2. *Generation of Marker:*
   Here, you have to write a script which can create a marker belonging to 'Original ArUco' dictionary with a specific ID. The ID will be given by the user. After creating the marker, add a white border around it (thickness of the border upto you). The dimension of the marker generated should be 400x400 (excluding the white border).

NOTE:

1. Please note that, in your final code, nowhere ArUco library should be used. Code found using it will not be evaluated.
2. The code should be as much as generic as possible. Think from all aspects. Since a user is involved for providing the data, keep in mind that you can't trust what value user will provide. Hence validate the data given by user first before blindly processing it.
3. Submission instruction will be the same. Strictly follow it.
4. If you want you can try implementing other dictionaries, but this is your wish. No excuses will be entertained if you fail to submit the main task because you went on implementing other dictionaries.