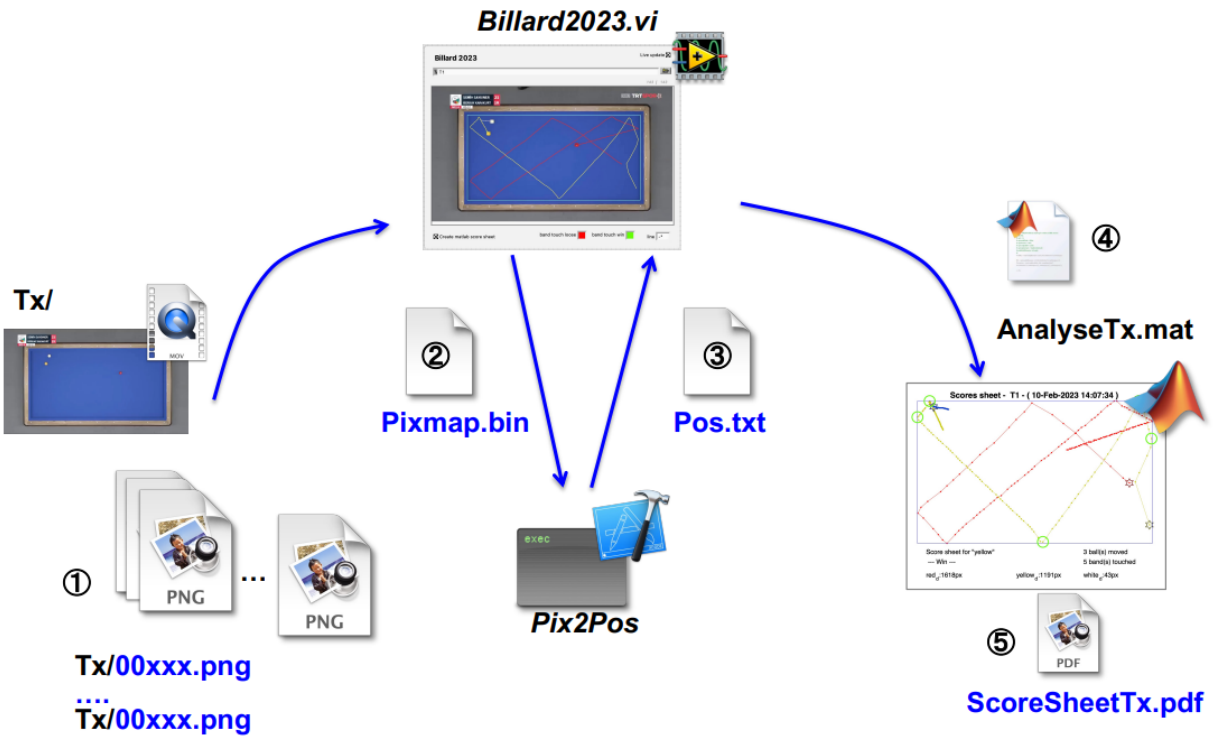
École Polytechnique Fédérale de Lausanne

ME-213 Programmation pour ingénieur

Date de rendu : 12 Juin, 2023



Roy TURK 34 55 73
Selim SHERIF 34 60 35

Introduction

The goal of Billiard 2023 is to analyze a part of a Carom's billiard game and determine whether the current player has won the game or not. The project is built using three different environments: VS code, MATLAB and LabVIEW. In this documentation, we will give a description of each part.

Pix2Pos.c

The C program (`Pix2Pos.c`) was written on Visual Studio Code and compiled on Windows (GCC). Its purpose is to analyse each image and determine the X and Y coordinates and the score of each ball.

We take as an input the `Pixmap.bin` file and create the `Pos.txt` file that contains the mentioned information for each ball. The program also checks for different types of errors and warnings that will be displayed in LabVIEW.

We will focus on all the functions used and give a brief explanation of how they work.

```c
void HEXtoRGB(unsigned int PIX, struct Pixel *RGB);
```

This function extracts each pixel in the `Pixmap.bin` file using bit-wise operators. We use in particular right shifting with `>>` and masking with `&` to end up with the corresponding RGB values for each pixel.

```c
int RangeTest(unsigned int PIX, struct Range Color);
```

The function calls `HEXtoRGB` then tests if the RGB values of the given pixel are within the imposed color ranges. It returns 1 if this true and 0 if else.

```c
void SquareScore(int Square_Y, int Square_X, int Width, unsigned int *Pixel,
                 struct Range Red, struct Range Yellow, struct Range White,
                 int *RedScore_ptr, int *YellowScore_ptr, int *WhiteScore_ptr,
                 int BallDiameter);
```

The void function uses a nested `for` loop to scan the received array and add to the square score by calling `RangeTest`. It is used in the `BallLocator` function.

```c
void BallLocator(int Lmin, int Lmax, int Cmin, int Cmax, int BallDiameter,
                 int Width, unsigned int *Pix, struct Output *RedBall,
                 struct Output *YellowBall, struct Output *WhiteBall,
                 struct Range Red, struct Range Yellow, struct Range White,
                 struct Range Blue);
```

The function runs through all the elements using a nested `for` loop to find the coordinates of each ball and assign a score by calling the `SquareScore` function. The score is updated only if it is greater than the previous one and if it is greater than `BallminScore`. If the latter is not the case, the ball keeps its original values which were initialized to {-1,-1,0} in the `main`.

```c
int main(int argc, const char* argv[]);
```

The `main` function receives 29 arguments from LabVIEW indicating the table borders, the color ranges and the ball diameter. It opens the `Pixmap.bin` file and reads it in binary mode. The PNG image properties are first read in order to allocate memory for all the pixels. The `BallLocator` function is then called to find the X and Y coordinates and the score of each ball. Finally, the function opens and writes the `Pos.txt` file in the required format.

Throughout all the code, we test for warnings and errors and a specific code is returned depending on the type of error.

Errors

- return 1:
  - Incorrect number of arguments
  - Ball diameter not in interval [5,20]
  - Image width or height not in interval [100,1000]
- return 2:
  - Problem while opening or closing `Pixmap.bin`
  - EOF reached before reading PNG image properties
  - Not enough pixels in `Pixmap.bin`
- return 3:
  - Problem while opening or closing `Pos.txt`
  - Problem while writing in `Pos.txt`
- return 4:
  - Problem while allocating memory using `malloc`

Warnings

- Passed the pixel limit in `Pixmap.bin`
- Ball is missing or score is not high enough

The documentation in the `Pix2Pos.c` file gives a more detailed explanation of the functions, variables and structures used.

AnalyseTx.m

The MATLAB script is used to create the score-sheet as a .pdf file and determine how many balls and borders were hit during the round. The score-sheet shows which ball was hit first and if the game was won or not. The user can choose to modify several elements of the score-sheet to his liking (from LabVIEW), including the line and border styles, the border hit style and color (win or loss colors).

In an analogous way, we will give a quick overview of the functions used.

```
function [Xmin,Xmax,Ymin,Ymax] = GetFrame(Xr,Yr,Xy,Yy,Xw,Yw)
```

The function uses the ball coordinates to determine the frame of the billiard, this corresponds to the blue rectangle displayed in the score-sheet.

```
function PathLength = GetBallPathLength(X,Y)
```

Using the Pythagorean theorem, the total distance traveled in pixels by each ball is computed and displayed in the score-sheet. Here, we make the assumption that the small variations in the X and Y coordinates when the ball is not moving are negligible compared to the total distance traveled, therefore we use all the elements in the vector with no condition on the ball movement.

```
function [FirstMoveIdx,MoveDist] = GetFirstMoveIdx(X,Y,
        MoveDistPx)
```

The function determines the first index after which the ball started to move under the condition $d >$ MoveDistPx. It also returns the vector `MoveDist` containing the distances traveled, i.e. the segments.

```
function [IdxTouch] = GetTouchIdx(X,Y,Xmin, Xmax, Ymin, Ymax,
        BallBorderDist)
```

The function returns the indices corresponding to a border hit. This time, the condition for a touch is $d <$ BallBorderDist.

```
function [FirstBall,SecondBall,LastBall, NbBallsMoved] =
        GetBallMoveOrder(Xr, Yr, Xy, Yy, Xw, Yw, MoveDistPx)
```

The function is called to determine the number of balls that have moved and their order. If two balls start to move from the same index, the tie is broken by comparing the biggest initial distance traveled.

```
function [X,Y] = InterpolateNan(X,Y)
```

When a ball is missing, i.e. {-1,-1} coordinates are assigned, LabVIEW replaces them with NaN values. We use the `interp1` function to interpolate with the next valid value. We also make the assumption that the final value in a vector is not a NaN.

```
function [X,Y] = RemoveOutlier(X,Y)
```

When we have an outlier at the same index in both X and Y vectors, we use the function to replace the coordinates with the previous one. We choose to ignore the case where the outlier is in the first index and we make the assumption that outliers do not appear in groups.

Billiard2023.vi

The `Billiard2023.vi` is the main part of the project that connects everything together. In the front panel, the user enters the folder path that leads to the sequence of images and chooses the line and marker styles, border hit style and colors to be displayed in the final .pdf file. The errors generated from the C program or during execution are also on display in the front panel. The main VI is composed of sub-VIs that manage `Pix2Pos.c`, `AnalyseTx.m` and errors.

Upon execution, the VI starts by finding the table limits using `TableLimits.vi`. Then, we start a loop for each PNG image:

- `PNG.vi` gets the image properties, i.e. width and height.
- `PIXMAP.vi` creats the `Pixmap.bin` file using the Write to Binary File function in little endian.
- `PIX2POS.vi` is used to call the C program executable. In it, we find a sub-VI called `CommandLine.vi` that creates the command line by concatenating the table limits with the other arguments of the color ranges and ball diameter.
- `POS.vi` is the final step of the loop where the ball coordinates and scores are scanned into an array. Before finishing each loop, we test if a ball has {-1,-1} coordinates in order to replace them with NaN values.

If the user chooses to create the score-sheet, `MATLAB.vi` creates the MATLAB script by concatenating various elements such as the arrays of coordinates, the line and marker styles, the border hit style and colors and the main part of `AnalyseTx.m`. Three sub-VIs are implemented to ensure that the user enters valid line and marker styles and a valid border hit style. In the case where this is not true, by default we set the line style to (-), the marker style to (.), and the border hit style to (o). Finally, the MATLAB script is launched using the provided sub-VI `MP_LaunchMatlabScript3.vi`.

Errors

Throughout `Billiard.vi`, there are sub-VIs managing different types of errors:

- The folder is not found or empty
- Error opening chosen folder
- Folder name does not match the format `Tx`
- PNGs in folder do not match the format `%d.png`
- Error executing `Pix2Pos`
- Error in `Pix2Pos`