Using Image Processing to Identify and Score Darts thrown into a Dartboard

Student:  
Gergő Papp-Szentannai

UTCN

[gergo\_13@me.com](mailto:gergo_13@me.com)

Laboratory assistant:

Andra Petrovai

UTCN

[Andra.Petrovai@cs.utcluj.ro](mailto:Andra.Petrovai@cs.utcluj.ro)

*Abstract*—The following document is a report for an image processing project at the Technical University in Cluj-Napoca. The aim of the project is to get familiar with computer vision and image processing using OpenGL. The project examines the problem of keeping the score in a game of darts. Currently players must manually count the score or use less professional digital boards. In this paper the author presents a proposed solution for this problem.

Keywords—Image processing; OpenCV; thresholding; morphological processing; image segmentation.

# Introduction

## Topic context

“Darts” is a two-player-game, in which each player should score points. Each of the two players has three darts. Players throw their darts into a circular target, usually attached to a wall. The target is divided into 20 slices, each slice having a value from 1 to 20. A slice also has two special zones, that together make a double and triple ring. A dart thrown in these regions multiplies the score. There is also a third, smaller ring, called the “bull’s eye”, that is situated at the center of the circular target. To make a distinction between regions, normal regions (without multipliers) are usually colored black and white, while the multiplier regions (doubles and triples) are colored in green and red. These colors appear alternately, so that it is easy to decide in which zone a dart belongs to.

## Issues that should be resolved

During a match (that may be competitive or just at a hobby level), players must count their score at each round. Although this implies only basic mathematical operations (addition, subtraction and multiply), it can get hard to count the score. Also, keeping the score in a written form becomes unavoidable.

Some players even choose digital dart boards made of plastic that can detect and keep score using built-in sensors. These boards are however prone to break and their quality is bad. These types of boards are a solution for the score-keeping problem, however a proper solution should not reduce the quality of the game.



Figure 1

## Proposed objectives

A good solution should be one that does not affect the quality of the game but it can still count the score. This may be achieved by image processing on an already existing dart board.

The method presented in this paper takes images from a fixed position. The first image always captures the empty dart board to determine the regions of the board. Then at each round, the application detects the darts thrown into the board and counts the score.

The objectives can be summarized as the following tasks:

* To segment the image representing the dart board into regions – each region corresponds to a value;
* To detect the region where the darts are being thrown – and consequently to be able to count the score.

## Documentation structure

[I. Introduction 1](#_Toc484084859)

[A. Topic context 1](#_Toc484084860)

[B. Issues that should be resolved 1](#_Toc484084861)

[C. Proposed objectives 1](#_Toc484084862)

[D. Documentation structure 1](#_Toc484084863)

[II. RELATED WORK 2](#_Toc484084864)

[A. Existing literature 2](#_Toc484084865)

[B. Methods that can be applied 2](#_Toc484084866)

[C. Possible solutions 2](#_Toc484084867)

[III. PROPOSED SOLUTION 2](#_Toc484084868)

[A. Preprocessing 2](#_Toc484084869)

[B. Creating Mask 3](#_Toc484084870)

[C. Labeling Slices 3](#_Toc484084871)

[D. Assigning a specific value to each pixel 3](#_Toc484084872)

[IV. EXPERIMENTAL RESULTS 4](#_Toc484084873)

[V. CONCLUSIONS 4](#_Toc484084874)

# RELATED WORK

## Existing literature

The project with the same title as tis paper (and many others) was inspired by another image processing project at the Stanford University’s Digital Image Processing[[1]](#endnote-1) course. Several other projects can also be found on the Student-Proposed Projects section[[2]](#endnote-2).

The project entitled “Using Image Processing to Identify and Score Darts thrown into a Dartboard” belongs to Jacob D. Delaney and its report[[3]](#endnote-3) can also be found on the Student-Proposed Projects section.

They are several other projects that try to solve the “darts detection” problem, but few of them are well documented or open-source. Some of them are presented below:

1. *The Design and Implementation of an Automated Dartboard[[4]](#endnote-4)*: Dart detection using microphones, implemented on a microcontroller.
2. *Darts hit recognition using opencv[[5]](#endnote-5)*: A Scala implementation in OpenCV using a camera. The program detects the dart in real time.
3. *Object Recognition: The Darts Challenge[[6]](#endnote-6)*: Using Viola-Jones algorithm and Hough transform to detect an arbitrary dartboard.

The mentioned projects did not offer complete solutions to my initial problem, but they are certainly a good place to start researching.

I have not found other scientific papers that deal with the same problem as the one described in I.B, however it was easy to research on general problems dealing with image processing in OpenCV. The laboratory guides[[7]](#endnote-7) were also helpful in understanding the main concepts that are used in this project.

## Methods that can be applied

They are several methods that can be applied and most of them are already implemented in OpenCV. The following methods are widely used in pattern recognition problems:

* thresholding;
* morphological processing;
* noise filters (e.g. Gaussian filter);
* image segmentation;
* edge and line detection;

## Possible solutions

There are several possible implementations, but all of them must somehow implement the identification of each region (having a value and a multiplier). From my point of view this can be achieved in two possible ways:

1. Region identification by edge detection: This approach detects all edges, such that each region will be delimited by an edge. We can identify the value of each edge by examining its distance and orientation with respect to the center of the board.
2. Region identification by extracting color information: We can benefit from the fact, that a dart board has a predefined color scheme. Simple regions are alternating black and white, while the multipliers (doubles, triples and the bulls eye) are alternating red and green.

Both approaches will provide us with a point map, such that any given pixel will be mapped to a region. For example, pixel “X” could be mapped to the region “double 18”, “single bull’s eye” or “triple 7”.

The second part of the solution should deal with identifying the dart. This can be done in several ways, including pattern recognition, identifying a special color (if the dart is colored) or image comparison (between the empty board and the board that contains the image – the region that differs will represent the dart location).

From a technological point of view, it is mostly recommended to use OpenCV for image processing related algorithms. This can be done in several programming environments, such as Python, C++ and Java. Other implementations for C#, MATLAB and several other environments also exist.

# PROPOSED SOLUTION

My solution consists of four stages:

1. Preprocessing
2. Creating a Mask
3. Labeling Slices
4. Assigning a specific value to each pixel

After these stages, the program is ready to receive user input. On each click on the image, the program displays the score, based on the previous stages. Displaying the score is somewhat instantaneous, because all the required information is already available.

The first four stages are to be detailed in the following section:

## Preprocessing

In this stage, the colors of the original image (Figure 1) are split up, such that we can examine the green and red parts (Figure 2) of the image (you can notice that the multipliers on the board are always colored in red and green).

|  |  |
| --- | --- |
|  |  |

Figure 2

To eliminate some noise a Gaussian blur is first applied. The black & white image (Figure 3) is also computed using a calculated threshold.

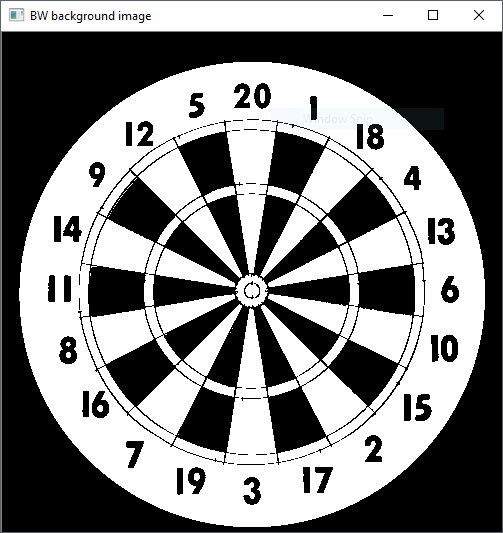


Figure 3

## Creating Mask

The Mask contains valuable information about the dart board that will be used in the final steps. We are interested in extracting as much information as possible, by applying morphological operations and thresholding on the image.

The mask computes the following information:

* White slices (Figure 4 – top right)
* Black slices
* Multipliers (Figure 4 – bottom left)
* Multiplier rings
* Board
* Miss (the complement of Board)
* Single
* Double (Figure 4 – bottom right)
* Triple
* Outer bull (Figure 4 – top left)
* Inner bull

Some examples are shown in Figure 4:

|  |  |
| --- | --- |
|  |  |
|  |  |

Figure 4

## Labeling Slices

Using the previously computed mask, we can identify each individual slice (each slice having a value between 1 and 20). We start by examining the White (Figure 4 – top right) and Black masks. We must erode the image to make sure the slices do not intersect. This is required because we apply an object labeling algorithm, that detects neighboring pixels, having the same value.

After applying the object labeling, we get a set of objects from each image (black slice and white slice). Because there might be many false objects, we compute the area (in pixels) of each object and discard those that are too small. The resulting set of objects coincide with the dart board’s slices, but are randomly labeled.

## Assigning a specific value to each pixel

In the previous step, we already identified each region. The value of the regions is however unknown. To detect the value of each region, we compute the center of each region.

The center of the board is already known, because it coincides with the center of the bull’s eye. We can compute the vector from the center of the board to the center of the region. We can use this direction of this vector to detect the position of the slice, w.r.t to the center of the board.

It is important to know, that the slices are predefined and they are arranged the same way for each dart board. By computing the angle of the vector, we can immediately get the value of the slice.

For example, if the slice is positioned at 90 degrees, we know that the value will be 20 (see Figure 1 or Figure 3). Similarly, we define an angle for each slice.

At this stage, we can know exactly the value of each pixel on the dart board.

# EXPERIMENTAL RESULTS

Results show, that the position of the camera influences the quality of the application the most. If the dart board is shown from one side, the circles are deformed and the algorithms are not that precise.

It seemed, that the quality of the images was not an issue, because with enough thresholding, it was easy to extract the most valuable information.

Currently the application detects most of the values, but it still needs some adjustments to perform better.

# CONCLUSIONS

In conclusion, I find this project very interesting and a very good place to learn about image processing and OpenCV

##### Reference

1. EE368/CS232: Digital Image Processing, Prof. Gordon Wetzstein [↑](#endnote-ref-1)
2. <http://web.stanford.edu/class/ee368/Project_Autumn_1516/index.html> [↑](#endnote-ref-2)
3. <http://web.stanford.edu/class/ee368/Project_Autumn_1516/Reports/Delaney.pdf> [↑](#endnote-ref-3)
4. <http://web.mit.edu/6.111/www/f2005/projects/mje_Project_Final_Report.pdf> [↑](#endnote-ref-4)
5. <https://github.com/vassdoki/opencv-darts>, László Vass [↑](#endnote-ref-5)
6. <https://github.com/louisditzel/OpenCV>, Ben Stokes, Louis Ditzel [↑](#endnote-ref-6)
7. <http://users.utcluj.ro/~andrapetrovai/ip.html> [↑](#endnote-ref-7)