# D3: Deep Dice Detector

*

1st Harsh Munshi *Hochschule Bonn-Rhein-Sieg*
Bonn, Germany
harsh.munshi@smail.inf.h-brs.de

**Abstract**

We present a two step approach for detecting the face value of the dice in any given die roll video sequence. The approach, named D3, employs a single shot object detector scheme to localize the dice in the auto-selected image from the video sequence at multiple scales and then uses blob detection to detect the value on the top face of the dice as well. In order to counter the perspective changes the algorithm also takes into consideration the location of the dice with respect to the camera and applies an intelligent cropping mechanism to avoid the side face pips getting detected as a part of the face value. The algorithm obtains an accuracy of 85.78% on the test dataset provided by the CV18 team.

**Index Terms**

Localization, Mobilenet SSD, Object Detection, Pip Detection, Image Processing, Blob Detection

## I. Introduction

Dice detection using cameras is a complex computer vision task. A die has faces and each face has a numbers represented by pips. The number on any die cannot exceed the total number of sides it has; so in a six sided die the numbers range from one to six. With variable lighting, die colors, foreign objects in the scene, various backgrounds, it is incredibly challanging to make an algorithm which can give a 100% result everytime is all conditions.

There have been many successful deployments of a computer vision based dice recognition system in gambling [1]. However it is only fair to point out that these systems work in controlled envrionment and most of the parameters are known.

There are two key components in a top face detection of dice in cluttered envrionments: The location of the dice and the reading of the top face in the localised dice. Recent works in genral purpose object detection using deep learning are groundbreaking in terms of both accuracy and ability to reproduce the results in dynamic envrionments. Object detectors like Singleshot MultiBox Detector (SSD) [2] and YOLO [3] consitute the state-of-the-art both in terms of optimized performace and accuracy.

However, object detection using deep learning demands high compute power and hence model compression techniques provides a key contribution in storing the models on the memory. Techniques like squeezeNet [4] and mobileNets [5] offer high model compressions while retaining the accuracy levels. It is only common to explore the opportunities to combined object detectors with model compression architectures as we see in the case of mobileNet-SSD [6].

## II. Deep Dice Detector

A two step aproach is proposed by D3 in order to recognise the dice given an image. Given a video sequence it is important to extract a stable frame for analysis.

**Data:** currentFrame, AverageFrame=None
**Result:** Frame For Analysis
initialization;
**while** *videoIsNotFinished* **do**
    read currentFrame;
    convert BGR2GRAY;
    apply BLUR;
    **if** *AverageFrame==None* **then**
        currentFrame=AverageFrame
    **else**
        weightedAverage(currentFrame, AverageFrame)
    **end**
    diff = difference(CurrentFrame, AverageFrame);
    thresh = threshold(diff);
    **if** *countNonZeros(thresh)* $\leq$ *50* **then**
        break;
    **else**
        continue;
    **end**
**end**

**Algorithm 1:** Frame Selection Algorithm

### *A. Frame Selection*

The frame select is handled by the preprocessing module. Algorithm 1 explains the algorithmic flow of the process.

It is important to note that since the initalization is done always on the first frame, that frame is passed; meaning it is not considered for evaluation. The following results give more clarity. As seen in Fig 1, the number of consequtive lower nonzero value frames gives us a clue for the required frame.

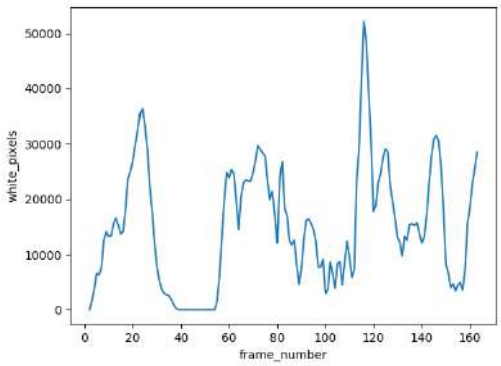


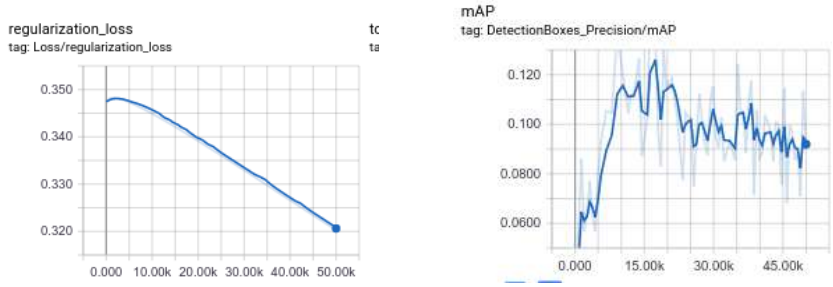

Fig. 1: Frame number v/s total white pixel

Identify applicable funding agency here. If none, delete this.

### B. Dice Localization

In order to make the detection loop robust, a deep learning based object detection schema is selected. The model is called mobileNet-SSD [6] which can scale very well on low end GPUs. Since our testbed was NVIDIA 1070 which is a medium range GPU, mobileNet can offer superior performance on the same.

In order to better train, 120 more videos were recorded along with the 20 given for the competition and each stable frame of those videos were annotated used labelImg. The annotations were done in pascal VOC [7] format. The classes used were ”dice”, ”sugarcube”, ”bottlecap” and ”foreign object” to better distinguish between a die and the noise. Post annotation the training was carried out using tensorflow object detection API [8] with a custom protoc file and it’s respective text file.




(a) Regularization loss (b) mAP

Fig. 2: Regularization loss and Mean Average Precision

Figure 2 shows the result of the training; regularization loss and mAP(mean average precision for the given test class). As we can see that there might be an overfit and hence it is not a good idea to have the detector detect the dice number directly.

Here are a few visualization of the detected dice. The model is pretty accurate and works well in almost all conditions.

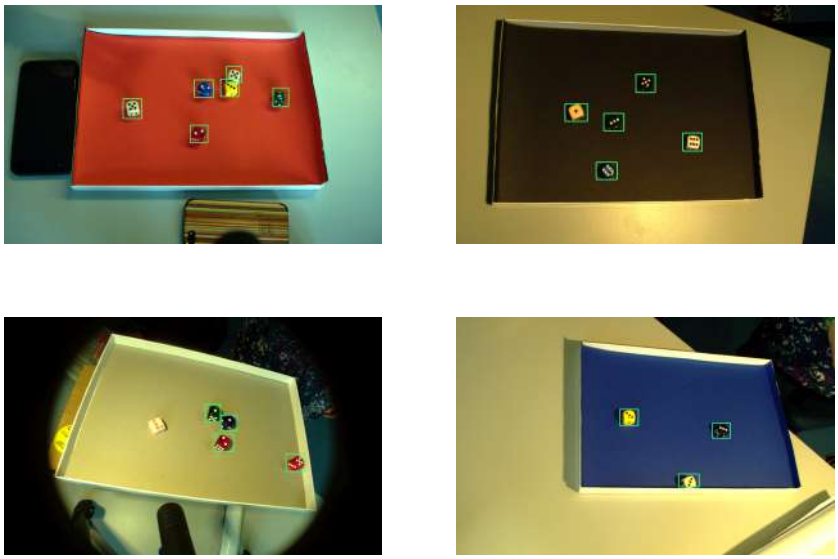


Fig. 3: Sample Results on Test Data

### C. Top Face Number Detection

After the localization we have individual die for analysis. In that case all we are interested in is the number of pips on the top face of the die. A blob detector which looks for convex shapes of particular circularity and intertial ratio will is used. However there is a drawback: what if the tray is tilted and the die is far off from the camera? In this case the side face of the die is usually detected as the top face based on the circularity. To counter that we propopse a solution which can be seen in algorithm 2.

```
Data: frame, croppedDie
Result: value
detector = blobDetector(color=255);
plane1, plane2, plane3, plane4 = divideFrame2fourCells(frame);
analyze:if croppedDie in lowerHalf then
    value = detector.apply()
else
    if croppedDie in plane1 then
        croppedDie.crop(0.2) ;
        value = detector.apply()
    else
        croppedDie.crop(0.4);
        value = detector.apply()
    end
end
if value == 0 then
    detector = blobDetector(color=0);
    goto analyze
end
return value;
```

**Algorithm 2:** Pip Counting Algorithm

The results look promising but needs more fine tuning. Here are a few example of the final output.

Fig. 4: Sample Final Results on Test Data

## III. Results and Analysis

In this section we explore the results on all the test data accross various parameters. We take into considertaion the three key factors: correct frame for analysis, dice localization and number of pips detection. Here is the table to give an overview of the same.

| Video Sequence | Correct Detections | Incorrect Detections | Correct Pip Values | Incorrect Pip Values |
|---|---|---|---|---|
| 2018-10-08@13-38-29.avi | 6 | 0 | 6 | 0 |
| 2018-10-08@13-54-00.avi | 5 | 0 | 5 | 0 |
| 2018-10-08@14-45-58.avi | 4 | 0 | 4 | 0 |
| 2018-10-08@14-55-15.avi | 6 | 0 | 5 | 1 |
| 2018-10-08@15-00-41.avi | 5 | 0 | 5 | 0 |
| 2018-10-08@15-14-43.avi | 5 | 0 | 5 | 0 |
| 2018-10-08@15-20-53.avi | 5 | 0 | 4 | 1 |
| 2018-10-08@15-26-37.avi | 6 | 0 | 5 | 1 |
| 2018-10-08@15-34-45.avi | 4 | 1 | 0 | 5 |
| 2018-10-08@15-41-24.avi | 5 | 0 | 2 | 3 |
| 2018-10-08@15-47-40.avi | 6 | 0 | 1 | 5 |
| 2018-10-08@15-51-36.avi | 6 | 0 | 4 | 2 |
| 2018-10-08@15-55-36.avi | 4 | 0 | 3 | 1 |
| 2018-10-08@16-03-10.avi | 4 | 0 | 3 | 1 |
| 2018-10-08@16-09-35.avi | 3 | 0 | 3 | 0 |
| 2018-10-08@16-13-24.avi | 5 | 0 | 2 | 4 |
| 2018-10-08@16-17-02.avi | 4 | 0 | 4 | 0 |
| 2018-10-08@16-20-32.avi | 3 | 0 | 1 | 2 |
| 2018-10-08@16-25-11.avi | 4 | 0 | 4 | 0 |
| 2018-10-08@13-54-00.avi | 4 | 0 | 3 | 1 |

We also have the following table to compute the accuracies,

| Total Dice | Correct Detections | Correct Readings | Score | Accuracy |
|---|---|---|---|---|
| 95 | 94 | 68 | 163 | 85.78 |

On the given dataset, D3 achieves 85.78% accuracy (based on the scoring criteria) and 98.94% detection accuracy.

## IV. Future Work

As we can see the main bottleneck for the recognition task is the pip detection. To counter this, we can have a much better image transformation algorithm which can check the orientation of the plate and apply affine transformation to all the boxes within the plate. This is always expose the top face of the die. Alternatively we can also train a classifier which can be used post detection which should be sufficient to achieve 90% accuracy or more in all the conditions.

# REFERENCES


[1] I. Lapanja, M. Mraz, N. Zimic, Computer vision based reliability control for electromechanical dice gambling machine, Industrial Technology, Proc. IEEE Int. Conf. 2000, 2, pp. 325-328.

[2] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In European conference on computer vision, pages 2137. Springer, 2016

[3] J. Redmon and A. Farhadi. Yolo9000: Better, faster, stronger. In Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on, pages 65176525. IEEE, 2017.

[4] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer. SqueezeNet: Alexnet-level accuracy with 50x fewer parameters and <0.5mb model size. arXiv:1602.07360, 2016.

[5] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam.Mobilenets: Efficient convolutional neural networks for mobile vision applications. CoRR, abs/1704.04861, 2017.

[6] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, Liang-Chieh Chen. MobileNetV2: Inverted Residuals and Linear Bottlenecks CVPR 2018.

[7] Everingham, M. and Eslami, S. M. A. and Van Gool, L. and Williams, C. K. I. and Winn, J. and Zisserman, A.The Pascal Visual Object Classes Challenge: A Retrospective, International Journal of Computer Vision, volume 111, 2015.

[8] Speed/accuracy trade-offs for modern convolutional object detectors. Huang J, Rathod V, Sun C, Zhu M, Korattikara A, Fathi A, Fischer I, Wojna Z, Song Y, Guadarrama S, Murphy K, CVPR 2017.