Artificial Intelligence and Computer Graphics

Improve the ball detection on the interactive pool project

Matthieu SEGUI

January 9, 2023

Supervisior
Clément Duhart

Introduction

Object detection is a huge field in computer vision and image processing. This technique allows to define and classify objects on an image. It can whether be done using classical approaches or with deep learning.

In order to track balls on a pool table, object detection algorithms are relevant. Those algorithms will use computer vision and deep learning processing in order to have fast and powerful pose estimation.

Computer vision algorithm are currently running on the pool (mainly image subtraction [1]). The current solution encounters latency and clusterization problems such as : sensitive to light changing, camera displacement and rotation. On top of that, the algorithm confuses balls with all round objects and closed fists.

We can separate detection algorithms in 2 main groups [2] :

- One-stage detectors
- Two-stage detectors

One-stage detectors are doing at the same time object classification and bounding-box regression. This performs without using pre-generated region proposals (candidate object bounding-boxes), e.g. YOLO [3] (You Only Look Once).

Two-stage detectors have two steps. First , a generation of region proposals, e.g. by selective search as in R-CNN and Fast R-CNN. Then an Object classification for each region proposal. Other processes can additionally be done such as bounding-box regression for refining the region proposals, binary-mask prediction etc. Two stages detectors achieve better accuracy but require more calculation ; inducing speed reduction.

The deep learning model required for the pool has to be very fast during inference time. Indeed, in order to have a solution running at 60 FPS, we have to track the balls in real-time. Model has to be robust to changing lights and camera movements. It also has to resist the human movement around the pool table during play time.

In order to track objects on a video stream, neural network has to compute on images. Convolutions will be the foundation of the neural network in order to reduce the input to features maps. The benefit is that the output of convolution can be input into another neural network (such as MLP). The result of the convolution are features maps that only contain mandatory information. RESNET neural network is a good way to benchmark a network [4]. This is affordable to implement and has good results with a quick inference time. U-Net models [5] are also an efficient way to make image segmentation. Those models are blind about the size of the input. However, U-net can be heavier in size and compute time but with more accuracy in the results.

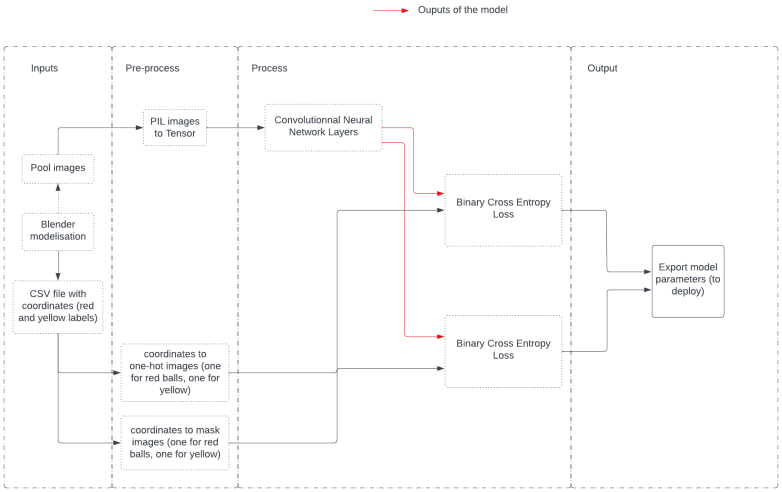


Figure 1: High level diagram of training pool ball detection. The origin of inputs is modelisation from Blender software

Pool ball detection

The project is based on a U-net model architecture [5]. This model will be deployed on the Interactive Pool Project [6]. The neural network has to be real-time.

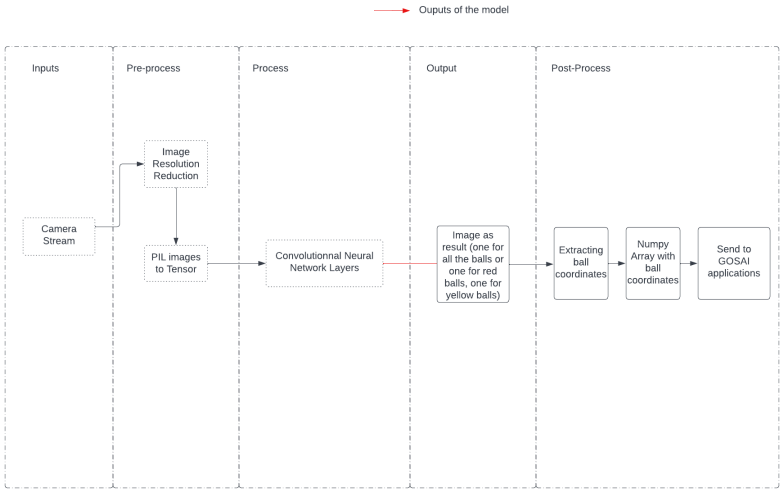


Figure 2: High level diagram of deployed pool ball detection. The origin of inputs is the camera. Outputs are used by several applications running on GOSAI.

The model will get the camera stream as input (figure 2). The output is currently a blank image with only dots represent the guessed center of the sphere (balls). This format of output allows to compare the result with the actual target.

A subjective evaluation of the AI is required. It will be showed and tested by random people of different age. Users have to be only focusing on playing on the pool, not annoyed by any latency or ball missing.

Overview

System architecture

The actual architecture of the neural network is based on a U-Net architecture [5]. This architecture is blind concerning the size of the input. The result will be at the same size as the input size. The conserving sizes allows the comparison with the target. If the model made transformations on the input image, it would be necessary to apply the same on the expected image (the target).

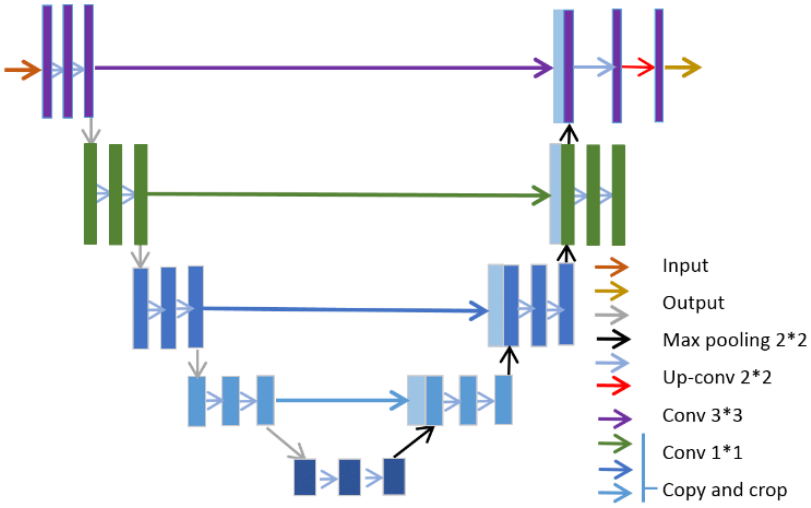
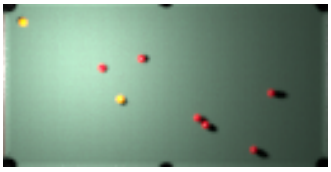


Figure 3: Diagram of the architecture of the U-Net.
credits : Ding and al. [7] - license : CC by 4.0

Generating Dataset

To train the actual U-net (or whatever the model is), the model needs data. And actually, this is shared with all the AI fields. Instead of taking picture of the pool and annotate each ball position, CAD softwares are more relevant. Data created from simulations are called "synthetic datas" [8]. Blender is an Open-source software. This software is able to reproduce the real ligths, shadows and reflection . Blender also allows to use python programming to automate the generation of data.



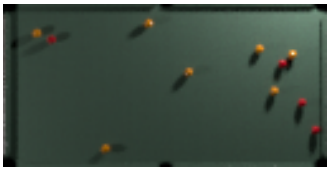


Figure 4: Preliminary outputs of Blender modelisations using CYCLES engine (without HDR). Resolution 128x64 pixels. Random number of balls, random light and balls coordinates

Pre-processing

As the dataset has been made for the model, pre-processing is pretty simple. The input image is transformed to tensor. The output coordinates are located on a matrix (making it an image) and transformed also to tensor.

Implementation

The AI is not yet deployed on the pool. However, the first results came out and can be compared to the objective.

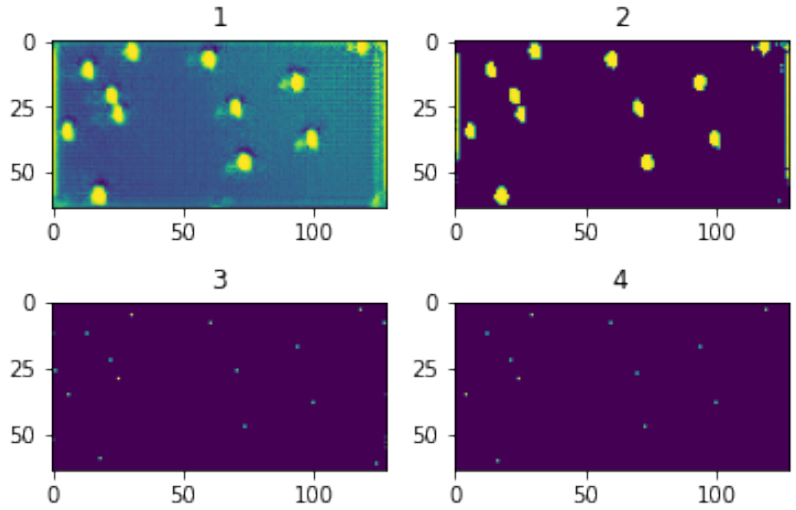


Figure 5: 1) Raw output of the model,
2) The output of the model after a threshold filter,
3) Extraction of the maximum value of the local cluster representing a ball,
4) The target

Looking at the results, the output (*1*) seems close to a pool with balls. Applying a threshold (*2*) and then a local maximum filter gives a result (*3*) that is comparable with the target (*4*).

To generate this output, the current U-Net is composed of 4 layers of down-convolution (Max pool + convolution) and 4 of up-convolution (refer to 3).

There is still some false positive when looking at the picture 3. This has to be fix. The model above has also 2 more problems. First, the inference time is too long for the real-time ball tracking application (Around 1 second for an iteration). Second, the model's size is too important (see figure 6). The problems are correlated.

```
Total params: 31,037,633
Trainable params: 31,037,633
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 12.00
Forward/backward pass size (MB): 16328.00
Params size (MB): 118.40
Estimated Total Size (MB): 16458.40
----------------------------------------------------------------
```

Figure 6: Summary of the trained model showed in figure 5. Size of the model is 16go.

Product Specific Success Criteria

Item	Description	Mark (total 20)
Dataset	Well made synthetic data with : - randomly generated balls - Close to reality lights - Noise - Extracted balls coordinates	3
AI	Build a working model	1
Performances	Correct inference time (at least 40 fps)	3
	Acceptable size (on computer with Nvidia 2080 GPU)	2
	Correct accuracy (Euclidian distance <3, GAP yet to compute)	3
Robustness	Resist to changing lights	2
	Resist to other round object	2
	Resist to other ball colors (other than yellow and red)	2
User study	Users not thinking about ball detection. Only enjoying activities of the pool	2

References

Here are the references in citation order.

[1] S.M. Desa and Q.A. Salih. 'Image Subtraction for Real Time Moving Object Extraction'. In: (2004), pp. 41–45. DOI: 10.1109/CGIV.2004.1323958 (cited on page 1).

[2] Zhengxia Zou et al. Object Detection in 20 Years: A Survey. May 2019 (cited on page 1).

[3] Joseph Redmon et al. 'You Only Look Once: Unified, Real-Time Object Detection'. In: (June 2016), pp. 779–788. DOI: 10.1109/CVPR.2016.91 (cited on page 1).

[4] Md Foysal Haque, Hye-Youn Lim, and Dae-Seong Kang. 'Object Detection Based on VGG with ResNet Network'. In: (Jan. 2019), pp. 1–3. DOI: 10.23919/ELINFOCOM.2019.8706476 (cited on page 1).

[5] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 'U-Net: Convolutional Networks for Biomedical Image Segmentation'. en. In: Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015. Ed. by Nassir Navab et al. Vol. 9351. Series Title: Lecture Notes in Computer Science. Cham: Springer International Publishing, 2015, pp. 234–241. DOI: 10.1007/978-3-319-24574-4_28. (Visited on 11/16/2022) (cited on pages 1, 3, 4).

[6] De Vinci Innovation Center - Interactive Pool. URL: https://dvic.devinci.fr/projects/educationnal-billard (visited on 11/07/2022) (cited on page 3).

[7] Yi Ding et al. 'A Stacked Multi-Connection Simple Reducing Net for Brain Tumor Segmentation'. In: IEEE Access PP (July 2019), pp. 1–1. DOI: 10.1109/ACCESS.2019.2926448 (cited on page 4).

[8] Matthieu SEGUI. How to generate synthetic data using Blender ? Oct. 2022 (cited on page 4).