Underwater object detection via Computer Vision technique

Kristopher H Jung
Shreyas Bhujbal, Lantao Liu
IURobotics, Vehicle Autonomy And Intelligence Lab

Purpose

The goal of the **RoboSub** competition is for an Autonomous Underwater Vehicle(AUV) to demonstrate its autonomy by completing underwater tasks. The state-of-art Computer Vision(CV) algorithm such as **YOLOv2** can operate at more than 90 Frames Per Seconds[1], which suffice the real-time object detection and tracking problem applying camera attached to the AUV. Therefore, I apply CV algorithm to detect underwater objects to complete underwater tasks.

Study Aims

Aim 1: Acquire, label, process and maintain underwater objects images datasets for YOLOv2 CV algorithm model.

Aim 2: Train YOLOv2 CV algorithm using GPU runtime environment in Google Colab Machine.

Aim 3: Mean Average Precision(mAP), and Average Training Loss verification over training time.

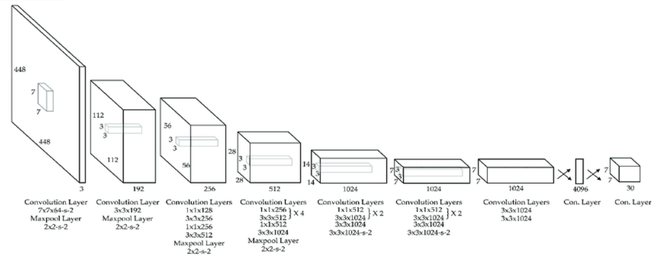
Aim 4: Applying custom object detection model along with decision feedback AI for completing various underwater tasks.

Experimental Design

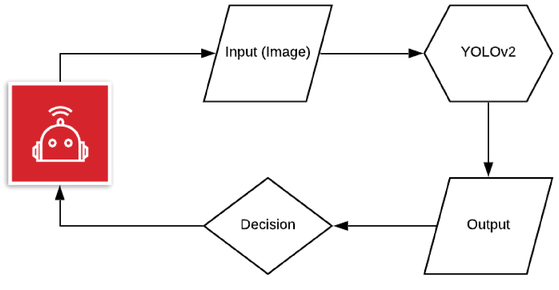
Training:

```
.
└── drive
    └── My Drive
        └── IURobotics
            ├── backup
            │   ├── yolo-obj_1000.weights
            │   └── yolo-obj_last.weights
            ├── bad.list
            ├── cuDNN
            │   └── cudnn-10.0-linux-x64-v7.5.0.56.tgz
            ├── darknet
            ├── darknet19_448.conv.23
            ├── img
            ├── mAP.PNG
            ├── obj.data
            ├── obj.names
            ├── script
            │   ├── Colab Notebooks
            │   │   └── darknet-example
            │   ├── process_unit.py
            │   ├── videoTest.py
            │   └── Yolo-Annotation-Tool-New-
            │       ├── Images
            │       │   └── monster
            │       ├── Labels
            │       │   └── monster
            │       ├── main.py
            │       ├── process.py
            │       └── README.md
            └── yolo-obj.cfg
```

YOLOv2:

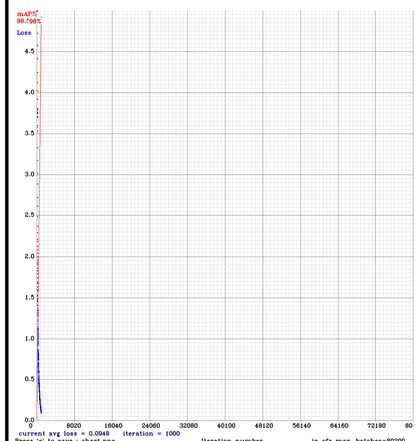


AUV Interaction:



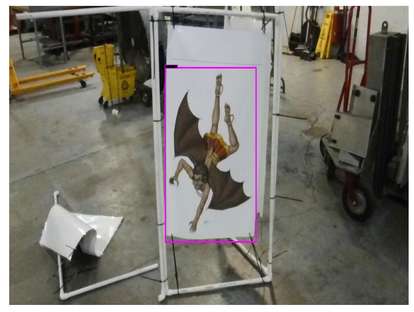
EXPERIMENTAL RESULTS

mAP and Training Loss verification:




```
 calculation mAP (mean average precision)...
256
 detections_count = 14823, unique_truth_count = 224
class_id = 0, name = monster, ap = 98.48%       (TP = 221, FP = 18)

 for thresh = 0.25, precision = 0.92, recall = 0.99, F1-score = 0.95
 for thresh = 0.25, TP = 221, FP = 18, FN = 3, average IoU = 75.10 %

 IoU threshold = 50 %, used Area-Under-Curve for each unique Recall
 mean average precision (mAP@0.50) = 0.984840, or 98.48 %
Total Detection Time: 33.000000 Seconds

Set -points flag:
 `-points 101` for MS COCO
 `-points 11` for PascalVOC 2007 (uncomment `difficult` in voc.data)
 `-points 0` (AUC) for ImageNet, PascalVOC 2010-2012, your custom dataset

 mean_average_precision (mAP@0.5) = 0.984840
Saving weights to backup//yolo-obj_1000.weights
Saving weights to backup//yolo-obj_last.weights
```

Detection model verification:



REFERENCES

[1] YOLO9000:Better, Faster, Stronger. Joseph Redmon and Ali Farhadi.

INDIANA UNIVERSITY
SCHOOL OF INFORMATICS, COMPUTING, AND ENGINEERING