# Finding Clipping Issues In Images taken from AC game


Presented in requirement of the Internship project

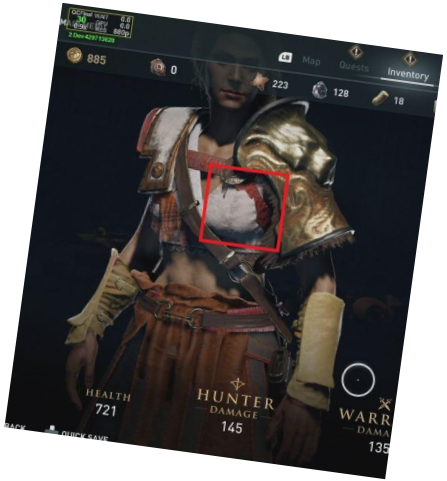
- Navnit Kumar, IIT Bombay ’21

# Defining Clipping :

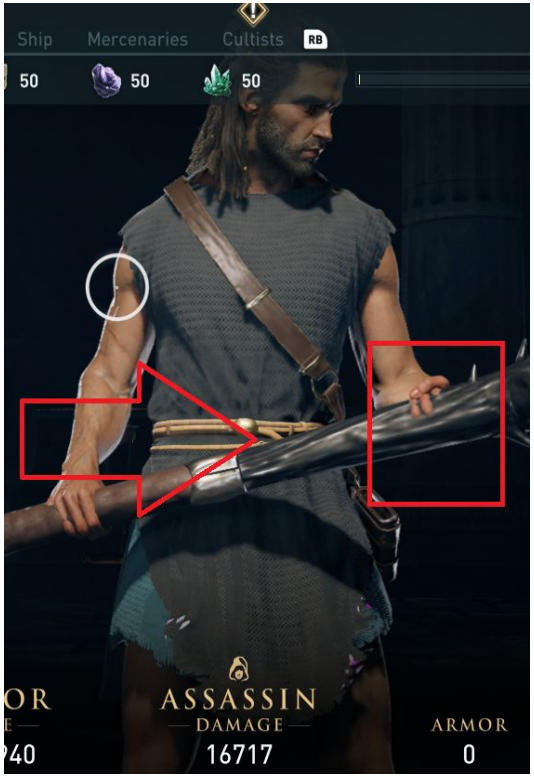
All the graphics in games are rendered by a GPU, which has **limited resource (memory)**.

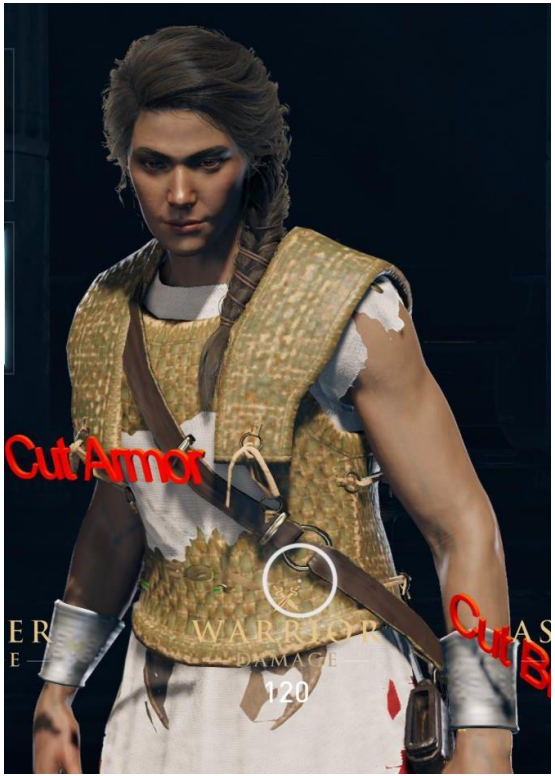
There are times when the GPU memory is not enough to render the graphic, in such cases it does resource optimization and presents scene to us. But in such cases,  graphics may not get correctly rendered.

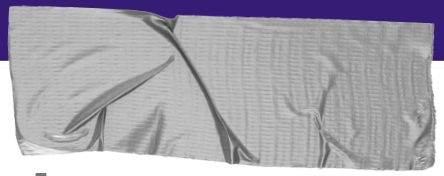
E.g. There are multiple layers of textures on a character and we observe a **mix of various layers** or **inner layers getting visible**.



# More examples of Clipping :



# 1. Intro

**Choosing one approach** to find the best and most-suited method to solve the clipping detection problem in images

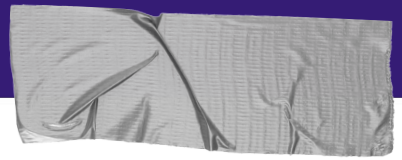
➔ **Image Processing**

Feature detection and description algorithms - **computationally intensive**, so difficult to achieve the speed of real-time performance. - differ in their capabilities and their performance is **highly input-dependent, no scalable output**

➔ **Deep Learning**

- **Object detection algorithms** RCNN, fast RCNN, faster RCNN and image segmentation techniques

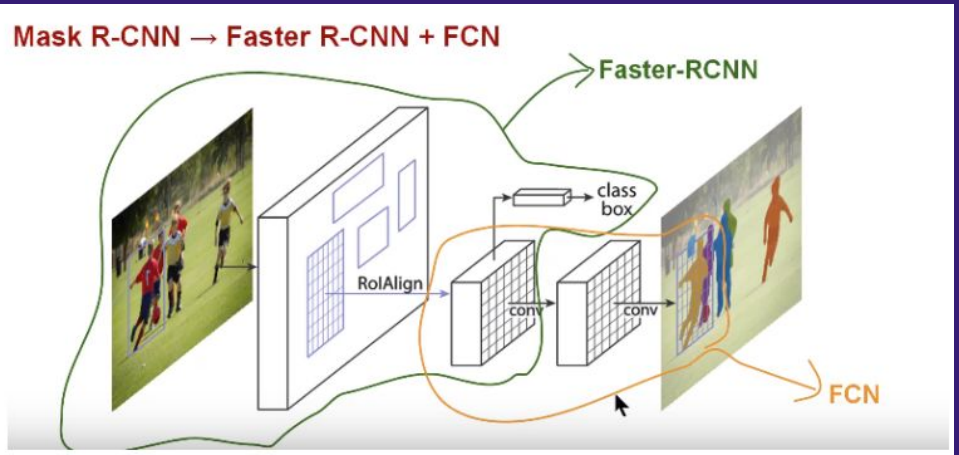
- narrowed down to **Mask RCNN** model



## Current Approach

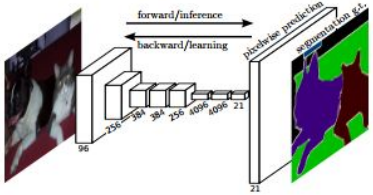
- **Deep Learning** based approach using Mask RCNN model.
- achieves ***Instance Segmentation***
- i.e. detects corresponding regions and generates mask (from pixel-wise classification of image) in bounding boxes
- **Faster** and more **accurate** as compared to other approaches even with less amounts of data (here, images)

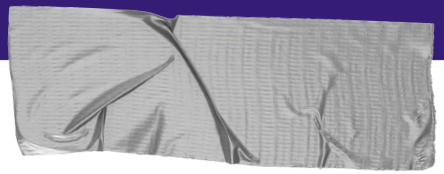
Source: [arxiv.org/MaskRCNN](arxiv.org/MaskRCNN)




Mask RCNN is a combination of Faster RCNN and FCN

Mask R-CNN is conceptually simple: Faster R-CNN has two outputs for each candidate object, a class label and a bounding-box offset; to this we add a third branch that outputs the object mask—which is a binary mask that indicates the pixels where the object is in the bounding box. But the additional mask output is distinct from the class and box outputs, requiring extraction of much finer spatial layout of an object. To do this Mask R-CNN uses the Fully Convolution Network (FCN) described below.





# STEPS INVOLVED TO DEVELOP THIS SOLUTION

**AIM** – To detect clipping issues/bugs in the images taken from AC game

## FLOW

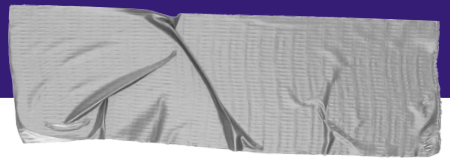
- **Data Preparation :** making a dataset of clipped images and split into train and test set

- **Data Annotation :** manually annotated clipped regions on raw data(samples) for training

- **Data Augmentation :** applied image augmentation to increase the size of dataset for training

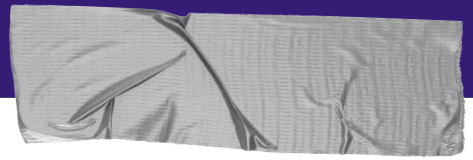
- **Training :** training the model on custom images and monitoring the loss for minimization after each epoch

- **Testing :** prediction on test samples (mix of issue and no issue samples) after training



## Preprocessing steps

- **Deep Learning** based solution requires large amounts of data for training
- But, only **30 images** in my dataset ?
- **Image Augmentation** to the rescue…
- Apply various **image transformations** viz. Flip, affine, rotate, stretch, etc. to generate more images
- Generated 8 transformed image from a single image. Increasing dataset size to **~250 images**

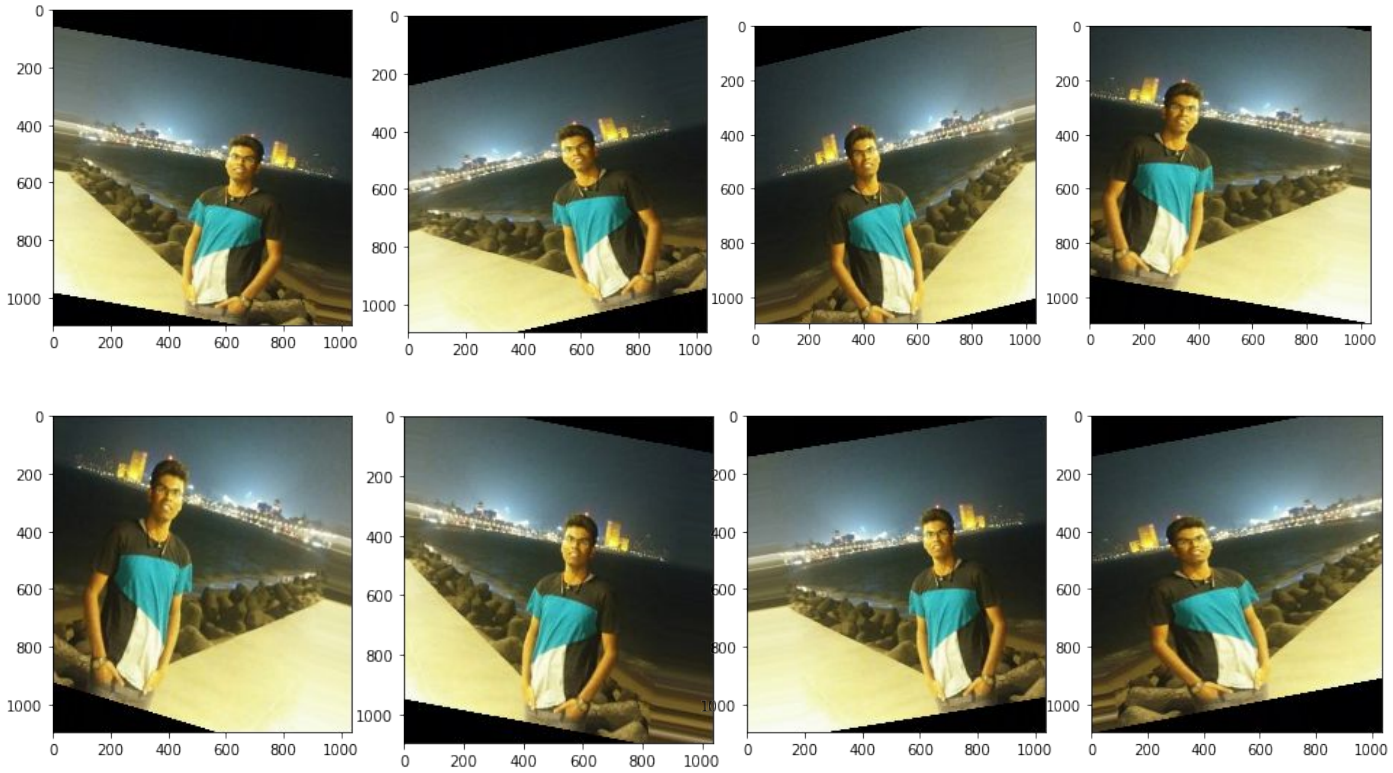


## Manual Annotation

- **Deep Learning** based solution requires annotations for training
- Used **VGG Image Annotator tool** to annotate initial training images for clipped regions before applying augmentation
- Using **JSON file** created, generated **binary masks**(1 : clipped, 0 : unclipped) for corresponding images
- Image Augmentation happens **on the fly,** hence, manual annotation is required only for initial 30 images in train set !

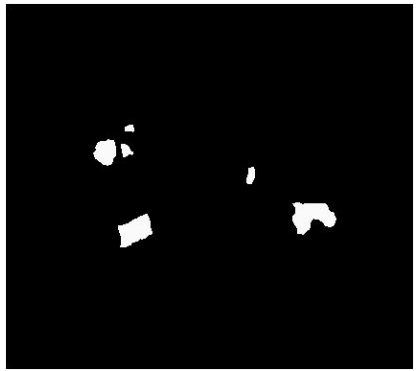
# Preprocessing steps (Generating more data)

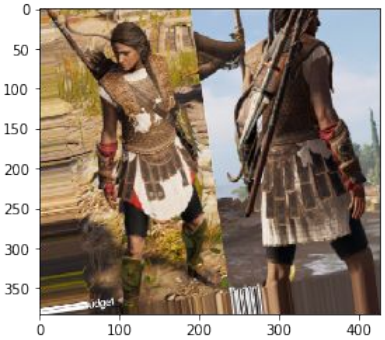
This is me ?

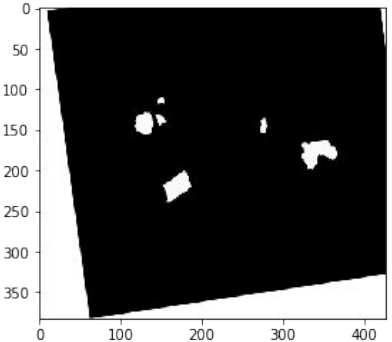


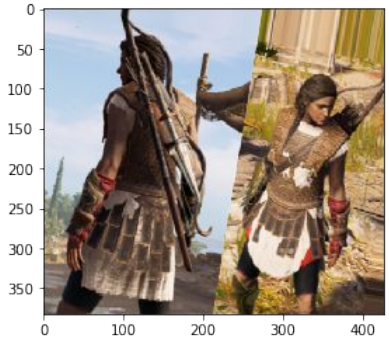
These are me augmented !!!

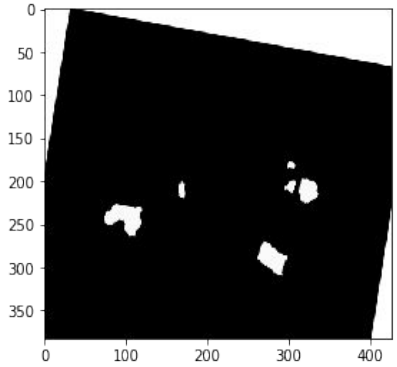
# Preprocessing steps (preprocessing with masks)

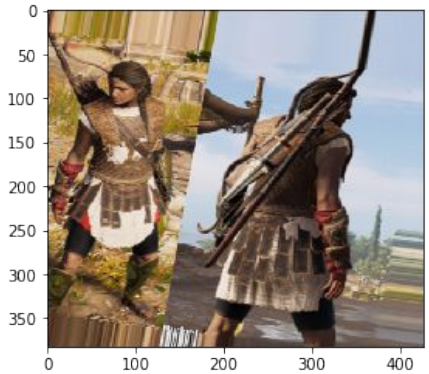


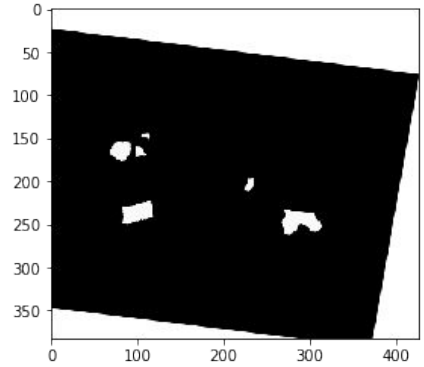






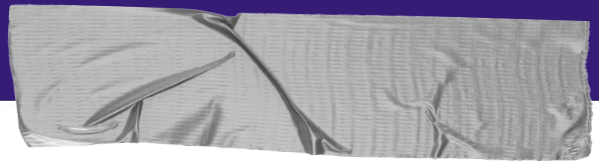






Original Image with mask

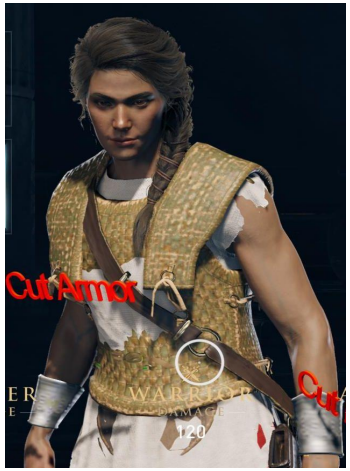
Augmented images with corresponding masks generated on the fly

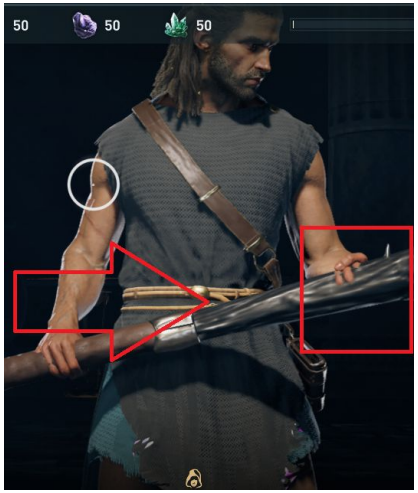


## Training model and Testing on  new images

- **Transfer learning** from coco/imagenet pretrained weights
- Model takes input image and generates corresponding binary masks, applies **same augmentation** to both, and **learns clipping features,**  by placing corresponding mask on the image after **minimizing loss** for detection.
- **Testing :** making prediction for clipping on test set . Good predictions, accurate to around **75 - 80%**
- **Postprocessing :** Mask accuracy could further be increased by using TTA
- **Test Time Augmentation :** apply augmentation to test images as well, before predictions and then **averaging predictions** from all augmentated images to give predictions on that image

# Results :

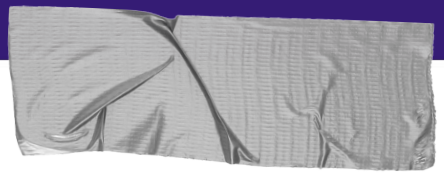

Cut Armor


50
50
50


clipped 0.905
clipped 0.898


clipped 0.888
clipped 0.967
clipped 0.860
clipped 0.853


50
50
50
clipped 0.865

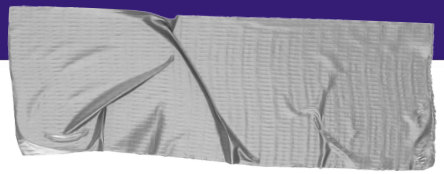


## NEXT STEPS

- Reduce False positives using improved TTA

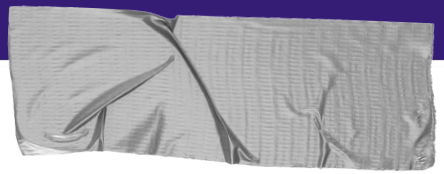
- Can work on Real Time rendering from video

- Can test the solution on multiple games (Crew2, FC etc.) using transfer learning with some training using custom game images



## Return On Investment & RESOURCE SAVING

- Model training with new images can take anywhere upto 1-2 days
- Final testing the real time video from source for bugs can be done overnight and detections are obtained in “clipped issues folder” with respective time stamp from video
- This tool can save upto few days to weeks of the game testers who can concentrate more on removing the bugs



## CHALLENGES FACED

- Local environment setup to run any DL model
- Reviewing through whole of codebase of MaskRCNN repo to make suitable changes
- Limited Training because GPU would fail to allocate required resource (limited memory)
- Brainstorming on selecting proper augmentation techniques to increase the model accuracy.

## LESSONS LEARNED

- Good data yields good results after processing.
- With deep learning we can automate processes but training it requires manual effort.
- Being regular and maintaining a record of what was achieved on a particular day goes a long way in completion of any task at hand.

# DEMO

# THANK YOU !