28th Summer School on Image Processing (SSIP 2020)

Project topic: Butterfly Recognition

https://github.com/khost95/SSIP-2020-Project-Butterfly/

Team members:

Goran Paulin – University of Rijeka, Croatia; goran.paulin@gmail.com / gp@kreativni.hr

Kristina Host - University of Rijeka, Croatia; kristinahost@gmail.com

Lien Le Phuong Nguyen - Szent Istvan University, Hungary; coni_love@yahoo.com

Matija Burić - University of Rijeka, Croatia; Matija.Buric@hep.hr

Jinsong Liu - Aalborg University, Denmark; jsliu91@gmail.com

July 2020.

Project description

One of the tasks as part of the 28th Summer School on Image Processing is to realize the project "Butterfly Recognition" through teamwork. The goal is to build a butterfly recognition framework which returns a list of potential matches ranked according to the similarity to a query image.

The problem involves several subtasks:

- coarse localization of the butterfly
- segmentation
  - hint: symmetry, color, shape might be representative features aiding segmentation
- appearance representation and matching
  - hint: analyzing the dataset with respect to discriminative features might be necessary
- evaluation
  - e.g. how recognition performance depends on the size of the training dataset, which species are similar to each other (confusion)

The given dataset: http://www.josiahwang.com/dataset/leedsbutterfly/

Datasets description

In order to build a butterfly recognition framework firstly is important to get along with the data you are dealing with.

The dataset LEEDS BUTTERFLY DATASET given by the SSIP can be found on http://www.josiahwang.com/dataset/leedsbutterfly/.

This dataset contains:

1. 832 images for ten butterfly categories (55-100 images per category) in PNG format, collected from Google Images and manually filtered. Each image is prefixed with the category ID of the butterfly (001, 002, ..., 010) followed by the sequence number of the image within the category.
   - The scientific (Latin) names of the butterfly categories are:
     001: Danaus plexippus (eng. Monarch)
     002: Heliconius charitonius (eng. Zebra Longwing)
     003: Heliconius erato (eng. Crimson-patched Longwing)
     004: Junonia coenia (eng. Common Buckeye)
     005: Lycaena phlaeas (eng. American Copper)
     006: Nymphalis antiopa (eng. Mourning Cloak)
     007: Papilio cresphontes (eng. Giant Swallowtail)
     008: Pieris rapae (eng. Cabbage White)
     009: Vanessa atalanta (eng. Red Admiral)
     010: Vanessa cardui (eng. Painted Lady)
2. Segmentation masks for each image in PNG format. Pixels with values 1 and 3 represent foreground pixels, and others background pixels.

3. Textual descriptions for each butterfly category, obtained from http://www.enature.com/fieldguides/. The files are named according to the category ID of the butterfly (001, 002, ... , 010).
    - Each text file contains three lines:
        Line 1: Scientific (Latin) name of the butterfly
        Line 2: Common (English) name of the butterfly
        Line 3: Textual description of the butterfly from eNature.com

In order to generalize the problem and make data augmentation we found various databases containing butterflies and have created one with synthetic data.

The dataset Butterfly Images-50 species that can be found on https://www.kaggle.com/gpiosenka/butterfly-images40-species/ consist of train, test and validation data set for 50 butterfly species. Only one species from this dataset (Zebra Longwing) can be found in the given one mentioned above. There is a total of 3558 images (train - 3158, validation - 200, train -200). The images are divided in 50 directories named after the belonging classes.

The third useful dataset Animals-10 that can be found on https://www.kaggle.com/alessiocorrado99/animals10. It contains about 28K medium quality animal images belonging to 10 categories: dog, cat, horse, spider, butterfly, chicken, sheep, cow, squirrel, elephant. All the images have been collected from "google images" and have been checked by human. In the dataset there are 2112 images of butterflies.

Also, we used 6 images of butterfly from Gábor Németh's Fotóalbum "Nature" that can be found on

http://users.atw.hu/ngweb/php/photos.php?album=Nature

Research and competition

There are many websites that explain how to recognize butterflies, but only a small amount of them have a search engine for it.

- https://butterfly-conservation.org/butterflies/identify-a-butterfly - identification by color, dimensions, markings
- http://gardenswithwings.com/identify-butterflies.html - identification by shape, open wing colors, closed wing colors, wingspan
- https://www.discoverlife.org/mp/20q?guide=Butterflies – identification by wing upper side main color, hindwing edge, family, distinctive color, forewing upper side distinctive color, forewing upper side distinctive pattern,...
- https://butterflyconservationsa.net.au/butterflies/identify/online-identification-tool/ identification by color, size and family

None of these include identification using images as input, only the fulfilled options.

With the development of Computer Vision more and more people are engaged around working with images and video, for example working on butterfly recognition. Here are some examples of Computer Vision used on butterflies.

- Butterfly detection and classification based on integrated YOLO algorithm https://www.groundai.com/project/butterfly-detection-and-classification-based-on-integrated-yolo-algorithm/1
- Butterfly Species Identification Using Convolutional Neural Network (CNN) - https://ieeexplore.ieee.org/abstract/document/8825031
- Butterfly Recognition Based on Faster R-CNN https://iopscience.iop.org/article/10.1088/1742-6596/1176/3/032048
- Classify butterfly images with deep learning in Keras https://towardsdatascience.com/classify-butterfly-images-with-deep-learning-in-keras-b3101fe0f98
- The Automatic Identification of Butterfly Species https://arxiv.org/abs/1803.06626
- Butterfly Species Recognition Using Artificial Neural Network https://www.researchgate.net/publication/324814938_Butterfly_Species_Recognition_Using_Artificial_Neural_Network

There are a few mobile apps that consider butterflies, most of them are designed like encyclopedias where you can search about the wanted species and read information about it, there is one for counting butterflies (if nearby you see a butterfly you can mark it on a map). Considering Butterfly recognition, we found some mobile application, the one worth mentioning:

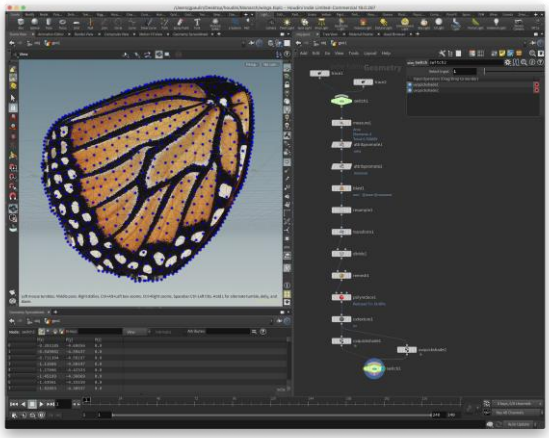
Name	**type**	**focus**	**detail**	**recognition**
Picture Insect – Insect Id Pro	Mobile, with payment, 7 days trial -IOS version	All type of insects	You take a picture, they give you the best match, and other possible matches with information about the species, if there is no match you can ask for help or suggest a name. There is also a community option when you can post your photos and ask others about the insect. The dataset is not visible, there is no search	It recognized the butterfly
Insect Identifier: Insect ID, AI Photo Camera	Mobile, free version with adds	Butterflies, Insect and Spiders	You can search for a species through the whole dataset or by category, you can take a photo, select the category and then they give you a name, if you want details there are retrieved from Wikipedia	No match
Insect Identifier App by Photo, Camera 2020	Mobile, free version with adds	Butterflies, Insect and Spiders	You take a picture, select a category, they give you the best match, and other possible matches with the confidence for each match, if you want details there are retrieved from Wikipedia. You can select a category in the feed and see what other people find. There is no search.	No match
Leps by Fieldguide	Mobile, free version, -IOS version -Login required	Moths and butterflies	You can search for species, and can publish a photo, while posting you can choose one of the species that are detected like a possible match.	No match

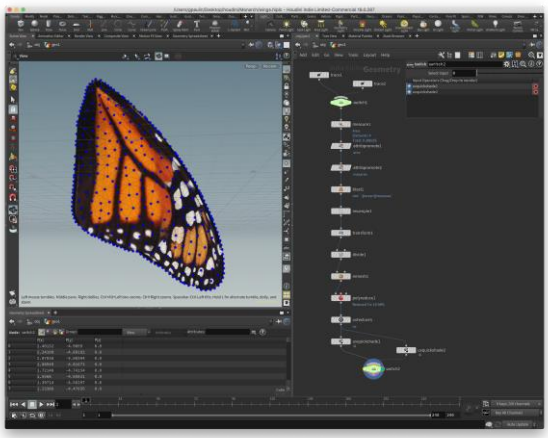
Background images dataset

The dataset consists of 80 background images taken during the summer school, on which created synthetic butterflies were inserted into to create synthesized data. The background images were taken in the daylight with android Samsung Galaxy s9 (12 MP, f/1.5-2.4, 26mm (wide), 1/2.55", 1.4µm, dual pixel PDAF, OIS).

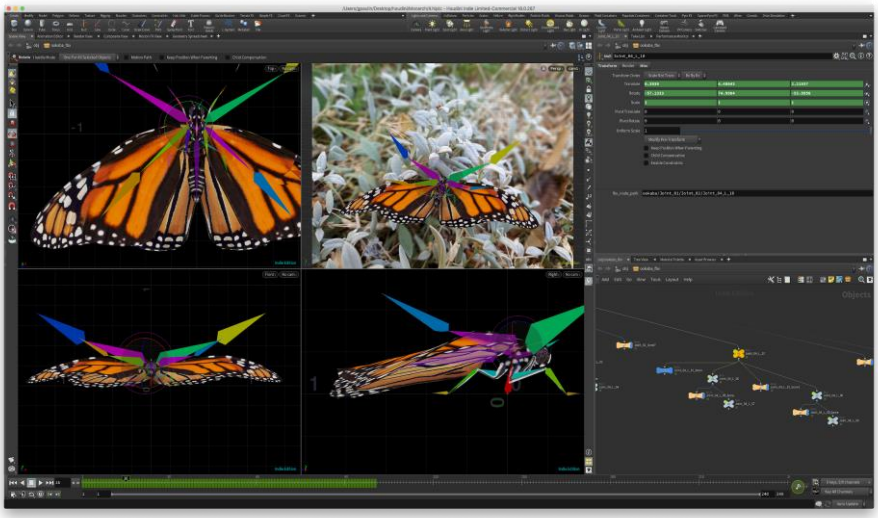
Synthesized data

Based on butterfly anatomy research, we created, in Houdini, a tool that takes two wing textures (since there are two different types of wings on a butterfly: a fore wing and a hind wing), converts them to 3D objects and could automatically attach them to the pre-animated butterfly body. This tool carefully traces input image, resamples it to create proper topology and then automatically reduces the number of polygons to conform to pre-defined vertices number, allowing both real-time use in game engines as well as traditional rendering. The final wing objects are automatically UV textured end exported.

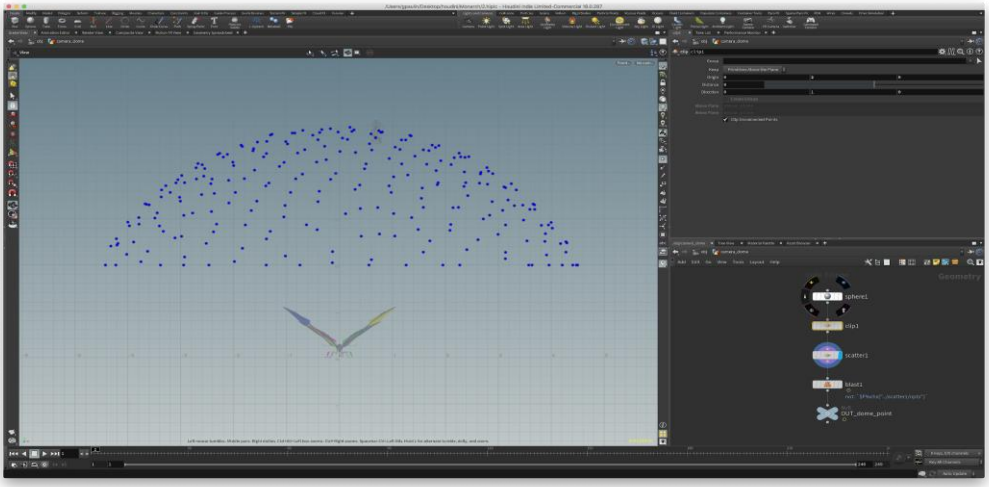




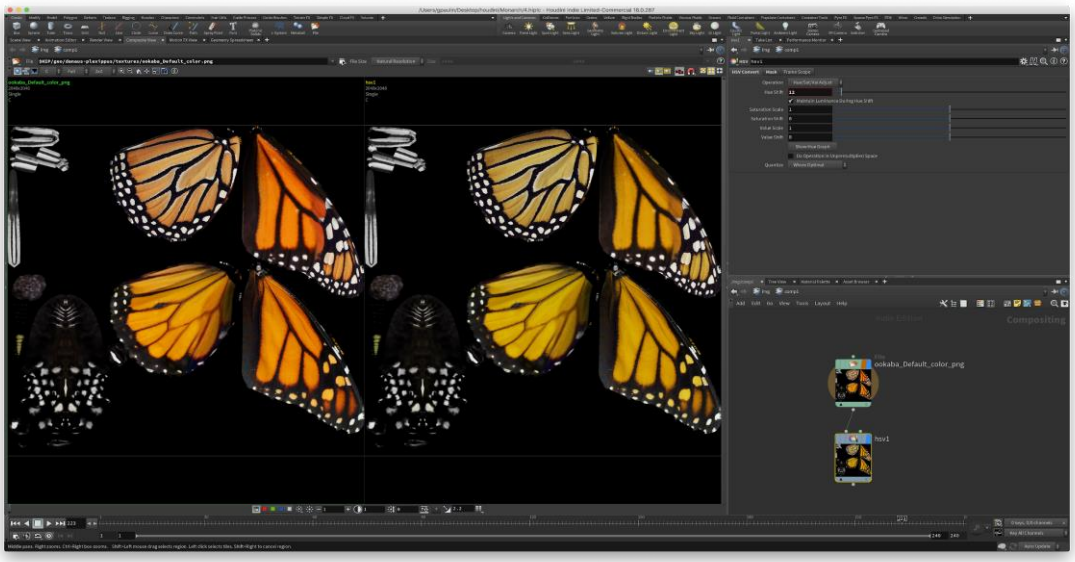
For the prototype, since we needed only a single butterfly, we skipped the automated attachment, of the wing to the body but it could be easily implemented.



Next, we sample random camera positions on the hemisphere, with a butterfly in the middle. During the tests, we reduced the hemisphere to about 1/3 of the sphere (upper part) because the camera angles less than about 20 degrees gave us unwanted results: very specific shots of butterflies that are not present in a real dataset.



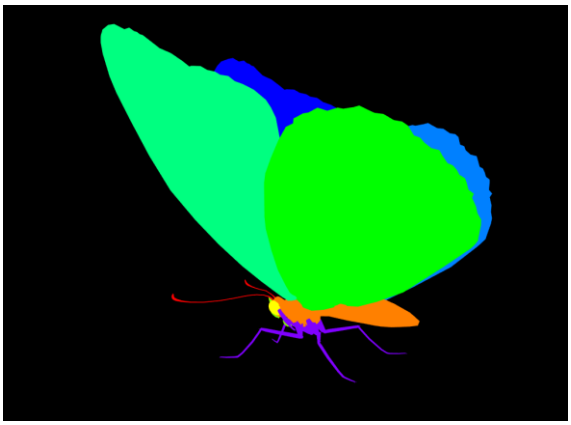
Before applying textures to the wings, we do simple domain randomization by varying the hue of the wing textures (between 0 and 12 degrees for Monarch). This gives us a wide enough range of colors to match the real dataset.



For the rendering, we use path tracing with variable numbers of ray samples, from 1 up to 9, to reduce the rendering time. We also use 3x3 pixel samples. During the rendering, we run the pre-defined animation of the entire butterfly (wings and the body moving) for the synth set 1 but for the synth set 2 we use static pose (no animation) for all the frames. We light the 3D butterfly using HDR image-based illumination, to match our background photos. Then we compose the butterfly image onto one of the 80 photographs that we created as a separate dataset. Our backgrounds dataset is also carefully crafted, matching the shallow depth of field usually seen in natural ecological photos, and is appropriate for compositing 3D insects on it, in order to achieve photorealism.

A binary mask is automatically created during the rendering process and we use OpenGL rendering for this, with 16x antialiasing (AA) samples, to speed up the generation.

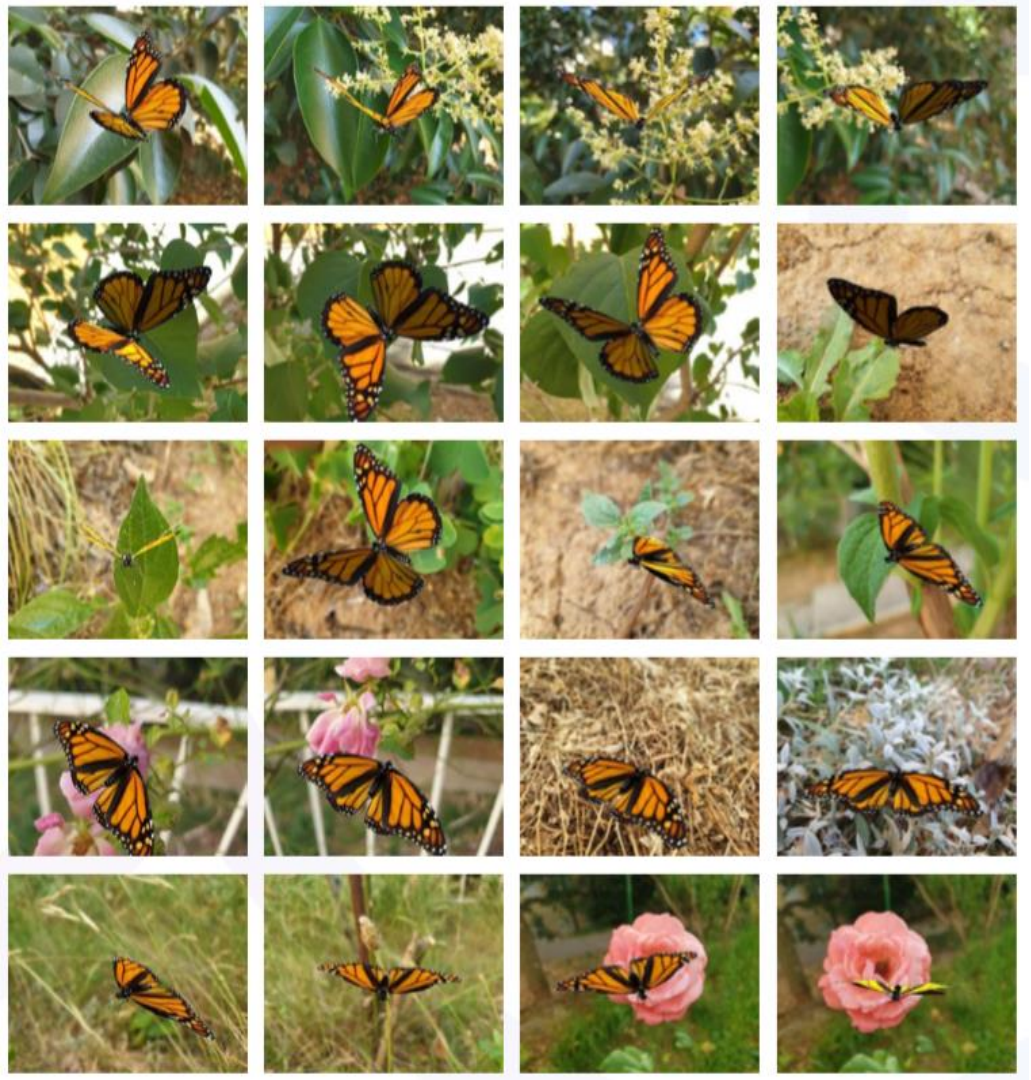
Since we have access to each part of the butterfly, we group them and we can automatically produce a segmentation mask of the parts which can then be used with the GAN approach to steer deformations of the butterfly body's components, if required.

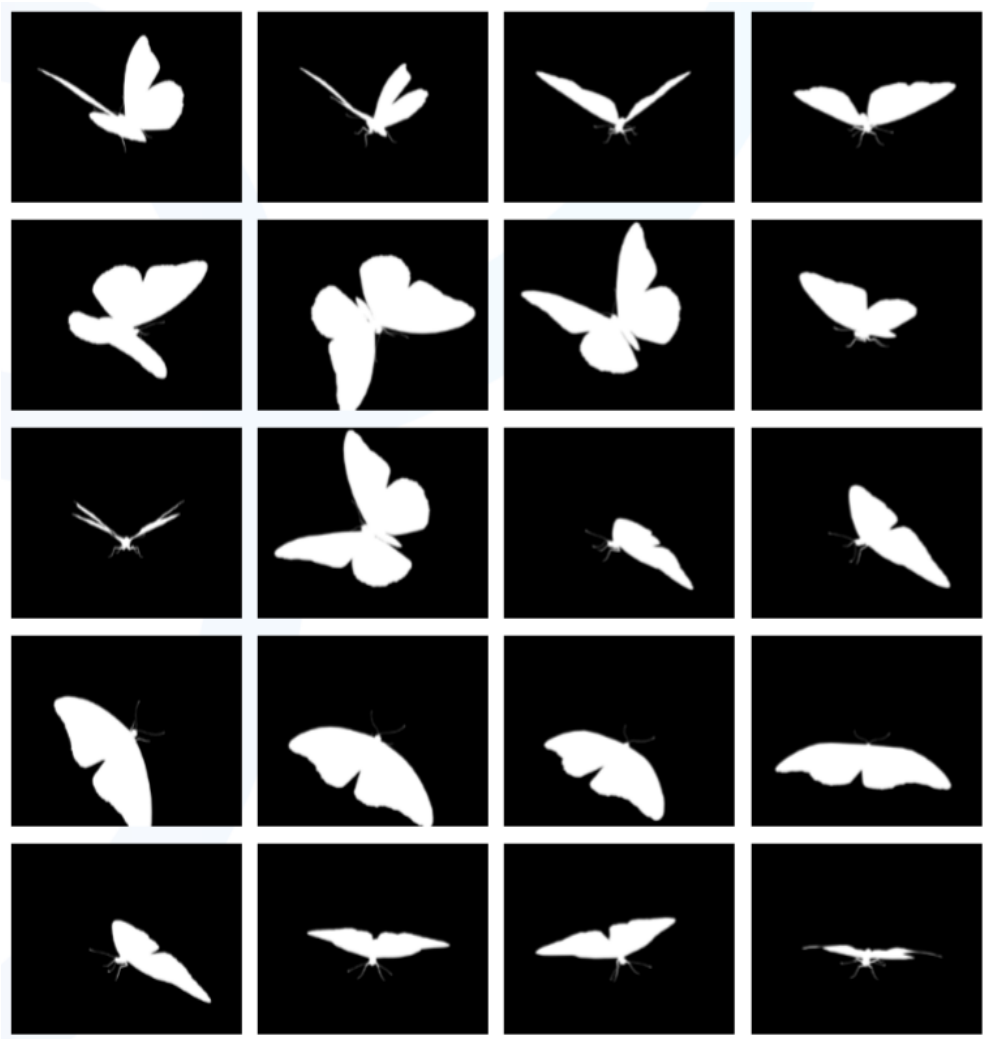


This way we ended up with two synthetic datasets for one butterfly class (Monarch), currently having 480 annotated synthetic images (240 in each set) for training a single class, compared to 82 Monarch images that we started with.

What we missed is the fact that Mask R-CNN uses a VIA annotation format. It's similar to COCO but differently formatted. Both COCO and VIA don’t use binary masks but coordinates of polylines.

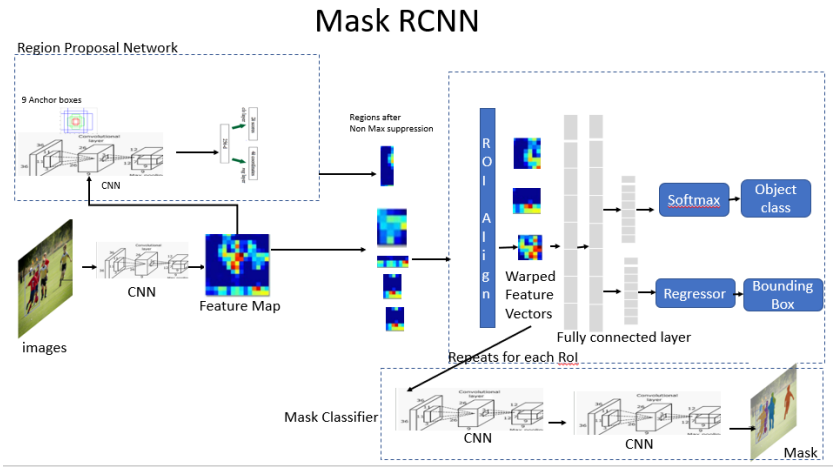
We couldn't find a binary mask to VIA converter, so we took a detour and ended up creating COCO annotations in Python, using pycococreator (https://github.com/waspinator/pycococreator/) and then converting them thru VIA web annotator (http://www.robots.ox.ac.uk/~vgg/software/via/) to VIA native format. Before running our binary masks thru the COCO converter, we had to extend the entire mask by 2 pixels because thin rendered shapes, such as butterfly's leg or antenna, prevented the creation of usable polylines-based masks. We did this extension using batch image processing in Adobe Photoshop.





Butterfly segmentation

For the main experiment, we used the well known Mask R-CNN in the Google Colab environment.



Programming was done using Python, Keras and TensorFlow. We had to take into account that the free version of Colab resets every few hours and therefore weights need to be backed up frequently enough. Processing was made using a Tesla k80 GPU which took us about an hour per 3 epochs.

We made our preliminary test using 20 train and 5 validation images just to see if it works or not. Results were looking promising but we knew that we needed more input images. Using a set of 240 training and 20 validation images we made progress.

The first real test was after 6 epochs and another after 15 epochs. We got mAP of 19% in the first test and in the second it got up to 26%. Then we changed the training dataset and trained for another 10 epochs, but this gave us degrading results so we decided to go back to previous data and continue training from 15 epoch again.

The final and best result we got was at 19 epochs with mAP at 27.7%. For evaluation tests, we used 12 images with different butterflies, single and multiple appearances, clearly presented and occluded. Images containing Monarch butterflies perform better as well as those that have nicely spread unobscured wings.

The script you can find on https://github.com/khost95/SSIP-2020-Project-Butterfly/blob/master/Image_Segmentation/ButterflyDetectionOnly_BEST19epoch.ipynb

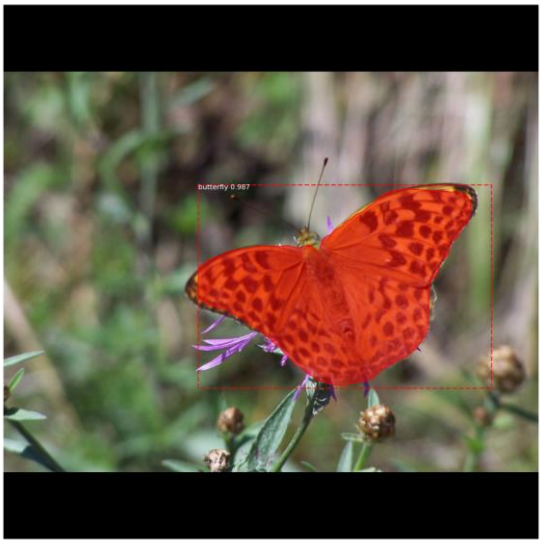
Predictions

Predictions

Predictions

Predictions

Predictions



Predictions

Novel approaches - Azure

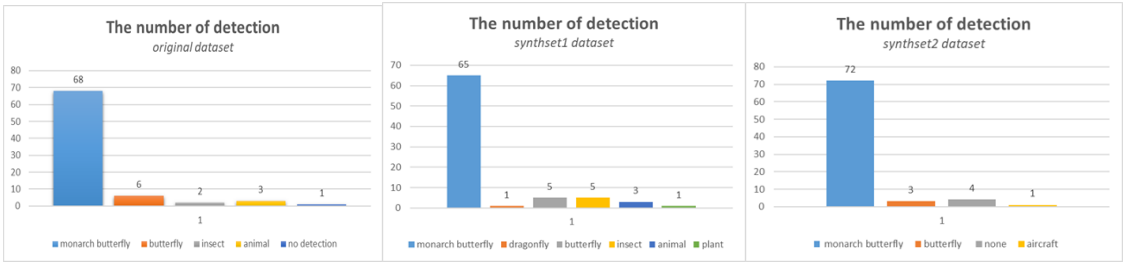
To evaluate the performance of a publicly available tool for image recognition and compare the results obtained on a given dataset and on the synthesized ones, we used Azure.

Azure cognitive services is an artificial intelligence “AI” service, based on cloud computing. It can process the picture and can recognize any common objects. In our project, we tested whether there is a butterfly among the recognized objects, but specifically if it could recognize the species. A python sample code was used and modified for batch processing. The Azure Computer Vision service is available as a web service (Software as a Service = SaaS). The application interface (API) is a RESTFUL web service, which is expecting picture data (up to 4 MB size) and answers in JSON format. Analysis with Microsoft Azure is not a novel method, but was intended to provide reference and opportunity to compare the team’s solution with commercial software. During the experiment, a free trial registration was used, which is limited to 5000 upload totally and 20 upload/min. There is another commercial solution at IBM called Watson Visual Recognition, but it did not work properly during the tests and we decided to use Azure.

We processed 820 images from the LEEDS BUTTERFLY DATASET. On the image below you can see an example where Azure recognized the species of the butterfly with the confidence value of 0.808. It is interesting how it shows parents classes and their confidence.

FEATURE NAME:	VALUE
Objects	[{ "rectangle": { "x": 170, "y": 148, "w": 180, "h": 171 }, "object": "monarch butterfly", "parent": { "object": "butterfly", "parent": { "object": "Moths and butterflies", "parent": { "object": "insect", "parent": { "object": "Invertebrate", "parent": { "object": "animal", "confidence": 0.849 }, "confidence": 0.827 }, "confidence": 0.826 }, "confidence": 0.823 }, "confidence": 0.823 }, "confidence": 0.808 }, { "rectangle": { "x": 106, "y": 244, "w": 306, "h": 158 }, "object": "flower", "parent": { "object": "plant", "confidence": 0.555 }, "confidence": 0.504 }]
Tags	[{ "name": "butterfly", "confidence": 0.990555167 }, { "name": "moths and butterflies", "confidence": 0.958591163 }, { "name": "insect", "confidence": 0.925485253 }, { "name": "invertebrate", "confidence": 0.915164232 }, { "name": "flower", "confidence":

Since there are 80 Monarch Butterfly images in this dataset, we took 80 images from each synthesized dataset with Monarch Butterfly and compared the results. In the left image it can be seen that Azure detected 68 times the species. The other times it recognized a parent class. There is also a case of non recognition. In the middle and right are shown the number of detection for our synthesized datasets.



In the table are shown some statistical information about the detection and the confidence obtained for Monarch butterfly recognition. It can be seen that with the Synthset2 we obtained the best result.

	Original dataset	Synthset1	Synthset2
Percentage of TP detection	85%	81,24%	90%
Mean confidence of the predicted MB	0,78	0,76	0,78
Mean total confidence for MB	0,67	0,61	0,70

The results obtained with Azure you can find on https://github.com/khost95/SSIP-2020-Project-Butterfly/tree/master/Azure_results

We also tested the specificity of the Azure system and uploaded a photo of bow tie, which has a shape similar to a butterfly. It was not recognized as a butterfly. The tested picture with the detection result is presented below.

FEATURE NAME:	VALUE
Objects	[{ "rectangle": { "x": 317, "y": 304, "w": 348, "h": 189 }, "object": "bow tie", "confidence": 0.546 }]
Tags	[{ "name": "person", "confidence": 0.999909639 }, { "name": "man", "confidence": 0.9996624 }, { "name": "necktie", "confidence": 0.997858 }, { "name": "wearing", "confidence": 0.9936049 }, { "name": "tie", "confidence": 0.9877319 }, { "name": "clothing", "confidence": 0.985422 }, { "name": "jacket", "confidence": 0.956833541 }, { "name": "collar", "confidence": 0.9256978 }, { "name": "shirt", "confidence": 0.9220699 }, { "name": "suit", "confidence": 0.9141288 }, { "name": "human face", "confidence": 0.74360764 }, { "name": "bow", "confidence": 0.632757366 }, { "name": "dress shirt", "confidence": 0.6320556 }, { "name": "bow tie", "confidence": 0.5900696 }, { "name": "beard", "confidence":

Image URL Submit Browse

Classification of Butterflies

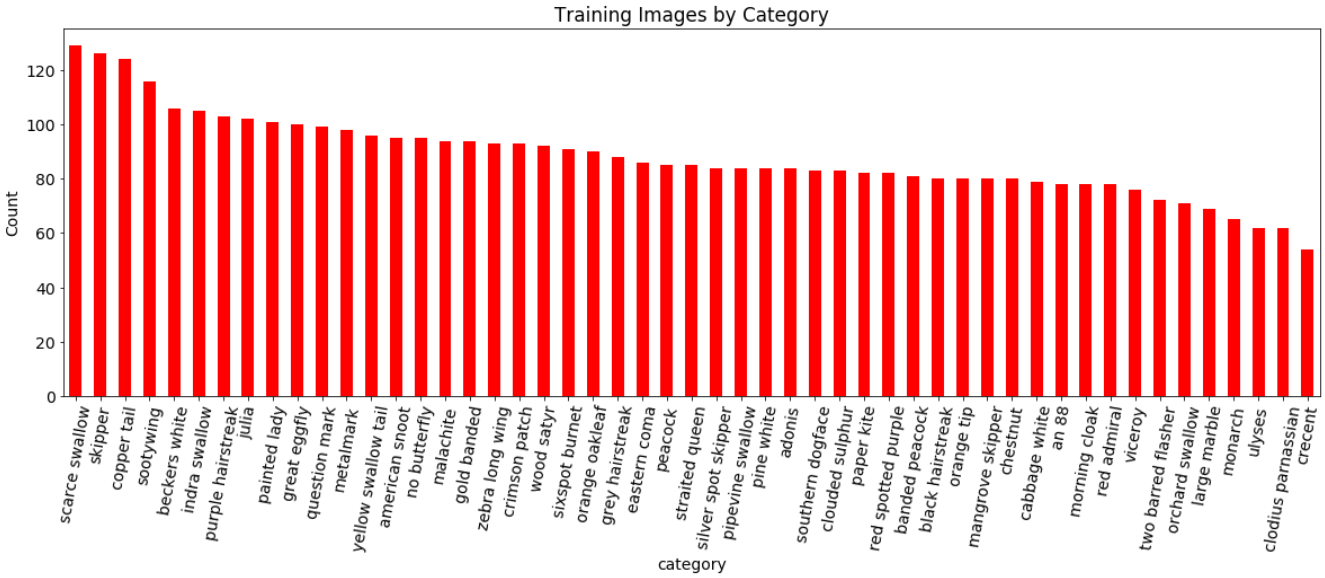
We implemented a 51-category classifier based on VGG16, where the 51 categories include 50 different species of butterflies and 1 category indicating all the other animals (like horse, dog, elephant, spider and so on).

You can see the jupyter notebook code in https://github.com/khost95/SSIP-2020-Project-Butterfly/blob/master/Butterfly_Classification/Butterfly%20Classification_vgg16.ipynb

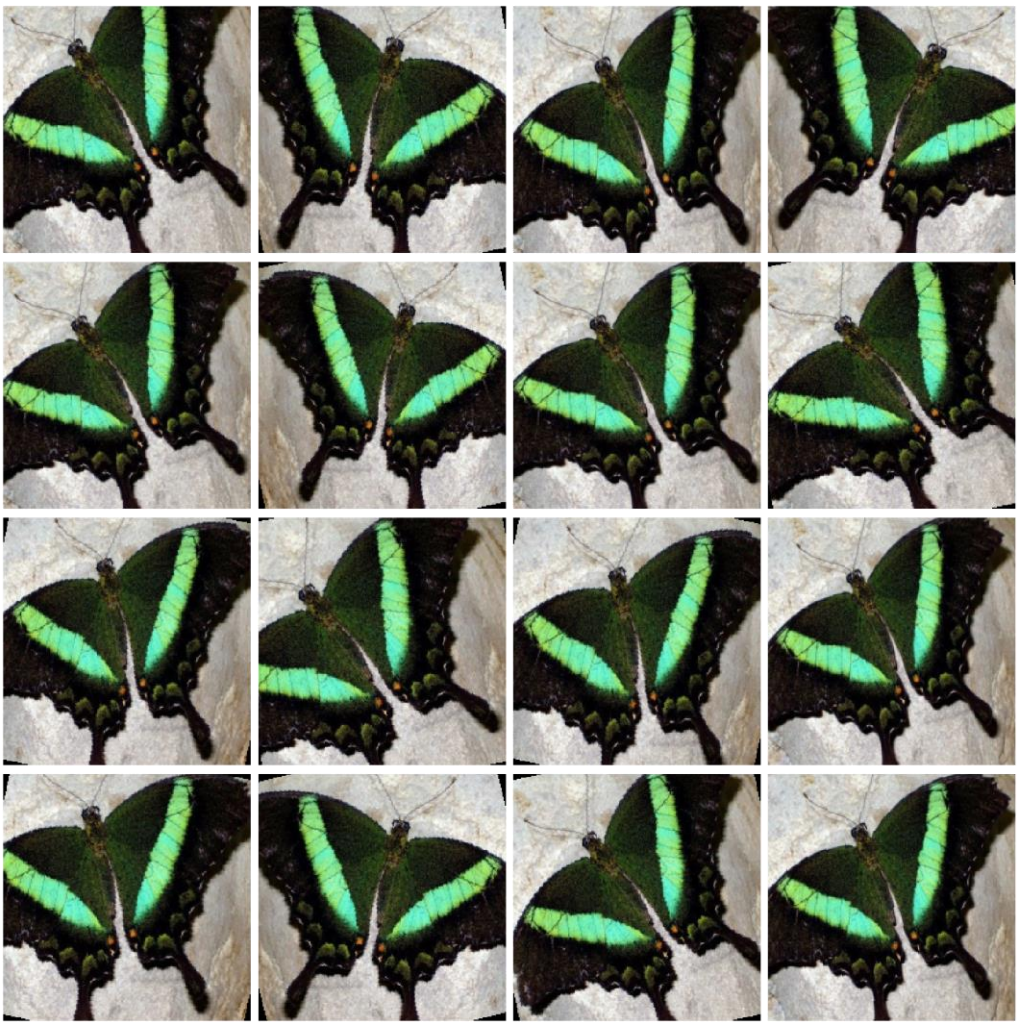
The hardware and software development platforms are 1 NVIDA GeForce RTX 2080Ti, Ubuntu 16.04 LTS, CUDA 9.2, Python 3.8 and Pytorch 1.5.1.

The dataset is a combination of the Butterfly Images-50 species dataset and Animal-10 dataset . As the number of training images in the first one is more than 5000, while its validation set and testing set have only around 250 images. The data partition is too unbalanced. Therefore after adding a new category called 'no butterfly' from the Animal Dataset, we rearrange the data partition into a training set with 4497 images, validation set with 519 images and testing set with 514 images.

Before the training, we first see the distribution of the training dataset and find that the number of images varies from 40 to 140 among different categories.



To avoid the reduced performance due to some categories with fewer images, data augmentation has been done for them, which includes randomly resizing and cropping, and flipping horizontally. Here is an example of an original image (train/banded peacock/banded peacock_1.jpg) and its augmented ones.



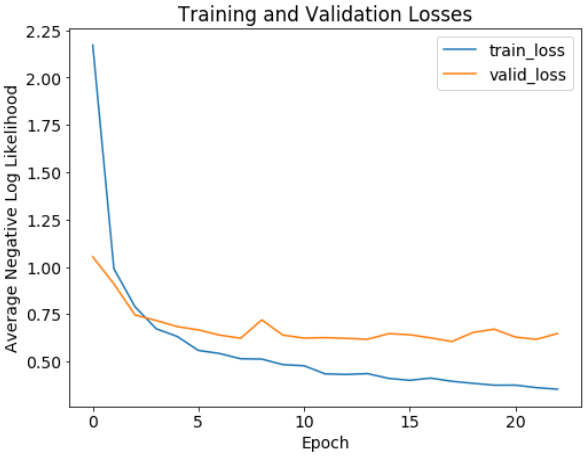
As this is a 51-category classification problem, a training set with 4497 images is really a small dataset even though data augmentation is used. Therefore, we seek the help of transfer learning with pretrained models of Imagenet dataset. And the exact model we use is VGG16, also ResNet50 can be an option for future work. Though VGG16 is not a state-of-the-art network, it has a very elegant architecture with all the 3*3 filters which is really a good choice for Deep-Learning beginners. From the below figure, you can see the network architecture of VGG16 in the column D.

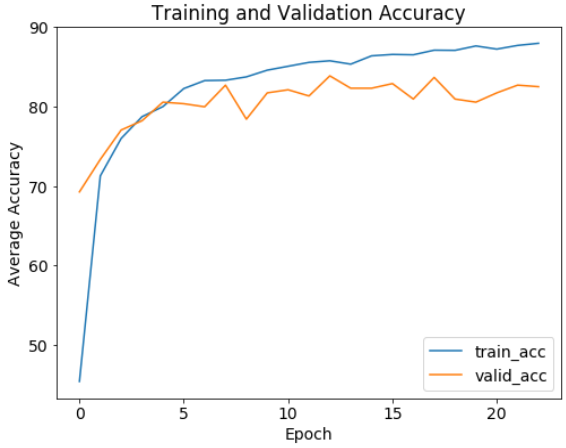
To use the pretrained model, we freeze the early layers of the network by preventing the gradient backpropagation to these layers. Then we change the last classification layer of the network from original 1000 outputs (1000 categories in Imagenet dataset) to only 51 outputs which correspond to our 51 categories.

As there are fully-connected layers in the network which require a fixed size of the input, the data feeding into the network is resized to 224*224. And also, a dropout of 0.2 is used to overcome potential overfitting problems.

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

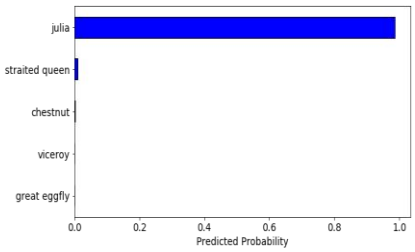
During training, early stopping strategy is utilized. That is, when the validation loss has not decreased for a number of epochs, the training phase stops so that the best checkpoint is saved for later use and the overfitting due to a long phase of training can be avoided. That's why our training ceases at the 22th epoch. The figures of training/validation loss and training/validation accuracy are shown below.

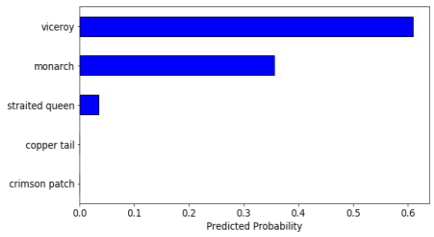


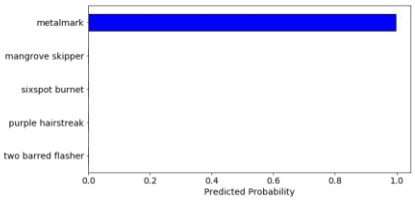


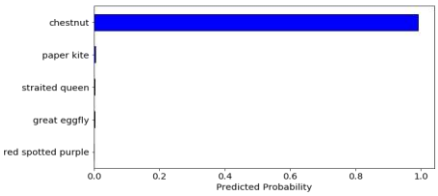
It seems that there is a gap between the validation accuracy and the training accuracy which may indicate a slightly overfitting that we attribute to the insufficiency of the training dataset.

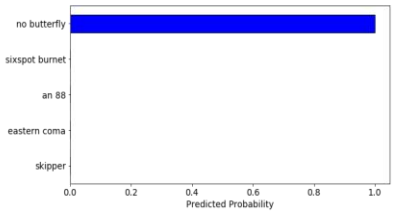
One function in the jupyter notebook can display the top-5 predictions for one image, which helps to see the performance of the trained network.

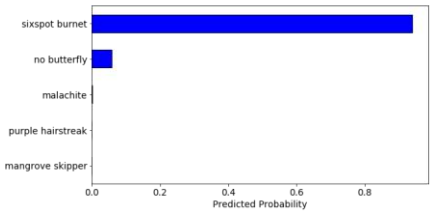










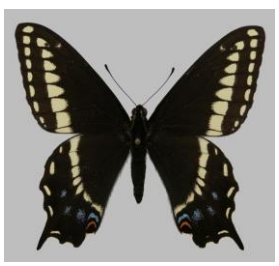


It seems that the above misclassification of the spider is due to the reason that the spider and the butterfly have similar sizes and shapes.

Because we also want to learn how to do training with a self-created checkpoint, we continue the training from the 22nd epoch checkpoint and stop it with another training 5 epochs.

Finally, the VGG16 achieves the top-1 accuracy and top-5 accuracy of 92.14% and 99.80%, respectively. This top-1 accuracy is still can be improved for two reasons:

1. The main reason is that the dataset is really limited and also difficult even for humans to recognize different species of butterflies. Like the figures below, the left one is a makachite butterfly and the right one is an indra swallow butterfly.



2. The network we use is not the state-of-the-art, for example, vgg16 model gets 28.41% top1 error and 9.62% top5 error on Imagenet, indicating its ability.

VGG results helped us to notice great similarities between Monarch and Viceroy classes that might cause problems in the future unless we don't handle them properly which we plan to.