Software as a Service for Shadow Detection and Removal from Images

Devshree Patel
AU1741075

Param Raval
AU1741083

Abstract—**Shadow Detection and removal is the process of enhance the computer vision applications including image segmentation, object recognition, object tracking etc. Detection and Removal of shadow from the images and videos can reduce the undesirable outcomes in the computer vision applications and algorithms. We make use of Stacked Conditional GANs trained on ISTD and present it as a web-based software application for commercial use.**

Index Terms—**shadow detection, Conditional GAN, computer vision**

I. Introduction

Detecting shadows, which occur naturally, can have a useful practical purpose in AR applications or scene study, where the lighting conditions play a major role in rendering realistic virtual objects and their shadows. Moreover, removal of shadows can help combat the performance degradation of many fundamental computer vision tasks such as object recognition, stereo, shape reconstruction, image segmentation and scene analysis. In regular life, applications with shadow removal can help in enhancing digital photography, improving aerial imaging as well as object tracking in videos.

II. Literature Review

Shadow detection and Shadow removal processes have helped in leveraging the performance of Computer Vision algorithms like Object Tracking, Object Recognition, Image Segmentation, etc. One of the first methods for Shadow detection was proposed by Shander et.al in 1999. In [3] object segmentation was used to detect the cast shadows using temporal integration of covered background regions. Following that, many different approaches have been introduced in the literature. In [2], authors found that shadows can be segregated using colors and used different intensity values for same background color.However, the approach fails when shadow and background share the same color and it takes more time for computation.

One of the first joint approach for detection and removal of shadows was proposed in [1]. In [4], authors develop an automatic framework using multiple ConvNets to detect and remove shadows in real world scenes. The model proved to be beneficial as it showed an ability to learn features at pixel level. Recent works like [5] and [6] make use of complex architectures involving Generative Adversarial Networks and other deep learning frameworks. These newly proposed methods formulate a pipeline to work on detection and removal simultaneously. Thus, reducing the overhead and fastening the process.

III. Implementation

A. Artificial Intelligence Service

Based on available approaches, we tried implementing two of them.

1) Instance Shadow Detection: In [5], to approach the problem of instance shadow detection, authors present three technical contributions in the base article. Firstly, the authors create a novel dataset — SOBA consisting of 1000 images and 3623 shadow-object association pairs, where each input image is accompanied by three instance masks. Secondly, the authors develop an end-to-end framework — LISA for predicting masks and boxes of object and shadow instances. The framework also builds boxes around shadow-object associations along with light directions. Later, ground truth instances of object and shadow are paired to match with the predicted shadow-object association and light-directions for producing output. Thirdly, the authors formulate an evaluation metric — SOAP for calculating the efficiency of the proposed model with existing baseline models.

2) ST-CGAN: Based on [6], ST-CGAN involves the use of two stacked Conditional GANs, each with a generator and a discriminator, such that it performs multi-task learning and does both shadow detection and removal. Figure 5 describes the internal functioning of the ST-CGAN architecture.
First (G,D) pair of the two stacked Conditional GANs, (G1, D1) take in the RGB shadow image and work on the shadow detection task. G1 and G2 work in tandem to defeat the discriminator pair (D1, D2) over the input-GT shadow-GT removal triplet of images.

This model has been trained over the benchmark Image Shadow Triplets Dataset (ISTD) for 1500 epochs. ISTD consists of images with shadow, shadow mask and the shadow-free image as a corresponding triplet. There are 1870 such image triplets over 135 distinct scenes- 1330 for training, 540 for testing.

Figure 4 shows how it is integrated as a cloud-based software application.

B. Cloud Services

1) List of Services: Following are the services implemented in the final application:

- User Login/Sign up
- Allow user to upload images from local computer
- Allow user to shuffle images from the given test images
- Three Tier-ed subscription plan:
  Free Tier: for the use of upto 5 images
  Tier 1, 2, 3: Planned billing based on bulk use
- Safe and easy-to-use portal for facilitating transactions

- Allow user to view their usage history

As a Software as a Service (or SaaS) application, it allows a Customisable Business Logic, Subscription based Billing, and Single Sign On.

2) Development of Services: In terms of approach to implementation and coding, the services listed above are developed as follows:

1. User Login and Sign-up:
- A new user can sign up from the main page by entering their email address, username, and a strong password. This data is sent via the Flask API and inserted into the MongoDB collection of user information.
- A existing user can perform a quick login just once during the current session by entering their username and password. The authentication function in Flask sends a filter request to MongoDB collection and checks if the response returned is empty. If it is empty then a user by that username or password was not found.

2. Allow user to upload images from local computer:
- Once the user logs in and selects a service, they have the option to either test ShadowSight from an image on their local computer or choose one of the demo images provided in the application.
- After the file is uploaded, it will be temporarily stored in a directory on the AWS EC2 Instance and displayed on the ShadowSight UI page. This temporary image will be shifted to the S3 bucket once the necessary prediction results have been generated and displayed on the ShadowSight page.
- The prediction file generated by the AI model will also be stored temporarily and displayed. The user can also download the displayed image file directly and efficiently into their local system. Both the images will be shifted to S3 once the current image-upload session is refreshed.
- Maximum file size of the image is restricted to 10MB.

3. Allow user to shuffle images from the given test images:
- In the second method provided to the user, they can choose an image from the grid displayed for them. These images are stored in the S3 bucket and are fetched when the user selects on in order to perform the prediction task.
- The image grid can also be randomly shuffled to display more of the images available to the user.
- Processing and Predictions are performed on these predefined images in real-time just like any other image uploaded by the user.

4. Three Tier-ed subscription plan:
- Free Tier: Each user would be allowed 5 trials in total on images. The count will be kept by checking the number images the user has tested up till then in the MongoDB collection. If the count exceeds 5 then they will be redirected to the payment plan selection page.
- Paid Tiers: User can choose one of the following 3 payment plans:
  Tier 1: \$5 for 5 more images
  Tier 2: \$10 for 20 images
  Tier 3: \$15 for 25 images
  Upon exceeding the plan limit, the current plan is automatically renewed and the user is charged for the renewed plan.

5. Safe and easy-to-use portal for facilitating transactions:
- Stripe: Stripe is an online payment processing service that provides free APIs to developers which can be integrated into such an application. Stripe in ShadowSight will accept payments from the user based on credit card details entered by them. They need not have a Stripe account in advance.

IV. Implementation of the Services

- From the *flask* library of Python, we use the *request* module to get the method type (GET or POST) and form responses from the HTML front-end. *rendertemplate*, *redirect*, and *urlfor* functions enable information transfer and transition among webpages or perform calls to Flask routes defined in the script.

- The function *uploadimage()* renders the template for the AI service and upon a POST method, receives the uploaded file using *request.files*.

- The *stripe* library is used to set the API keys and perform a transaction. After receiving the payment and card details, the *checkout()* function will utilize the Stripe API to create a charge to the user using *stripe.Charge.create()* function that accepts Stripe account details of the recipient and the amount. A temp customer (or payer) is created by *stripe.Customer.create()* before this.

V. Diagrams

In Appendix A, Figure 1 shows the UML use case diagram and Figure 2 is the sequence diagram for the user login and authentication process. Figure 3 shows the flowchart of the working of the ShadowSight application. Figure 4 is the System Architecture diagram of ShadowSight as a whole.

VI. Results

Figure 6 consists of reproduced results for shadow detection and removal using the Stacked Conditional Generative Adversarial networks framework proposed in [6]. The stacked adversarial components are able to preserve the global scene characteristics hierarchically, thus it leads to a fine-grained and natural recovery of shadow-free images. Also, ST-CGAN consistently improves the overall performances on both the detection and removal of shadows using joint learning . Figure 7 consists of reproduced results for shadow detection and removal using the LISA framework proposed in [5]. Instance shadow detection is efficient from general shadow detection techniques, as it finds associated objects with shadows altogether instead of finding a single mask for all shadows.

Conclusion

Thus, we have the results from the shadow detection and removal model when run locally. Further, we were able to simulate a local Heroku application over a standard regression model where given a specific input the model returned the predictions over a Flask app. Next step in the process will be to setup a database and deploy the app on Heroku.

References

[1] Zhang, W., Fang, X.Z., Yang, X.K., Wu, Q.M.J.: Moving cast shadows detection using ratio edge. IEEE Trans. Multimed. 9(6), 1202–1214 (2017)

[2] Golchin, M., Khalid, F., Abdullah, L., Davarpanah, S.H.: Shadow detection using color and edge information. J. Comput. Sci. 9, 1575–1588 (2013). https://doi.org/10.3844/jcssp.2013. 1575.1588

[3] Stander, J., Mech, R., Ostermann, J.: Detection of moving cast shadows for object segmentation. IEEE Trans. Multimed. 1(1), 65–76 (1999). https://doi.org/10.1109/6046. 748172

[4] Khan, S.H., Bennamoun, M., Sohel, F., Togneri, R.: Automatic shadow detection and removal from a single image. IEEE Trans. Pattern Anal. Mach. Intell. 38(3), 431–446 (2016)

[5] Wang, Tianyu, et al. "Instance shadow detection." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020.

[6] Wang, Jifeng, Xiang Li, and Jian Yang. "Stacked conditional generative adversarial networks for jointly learning shadow detection and shadow removal." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018.

Appendix A

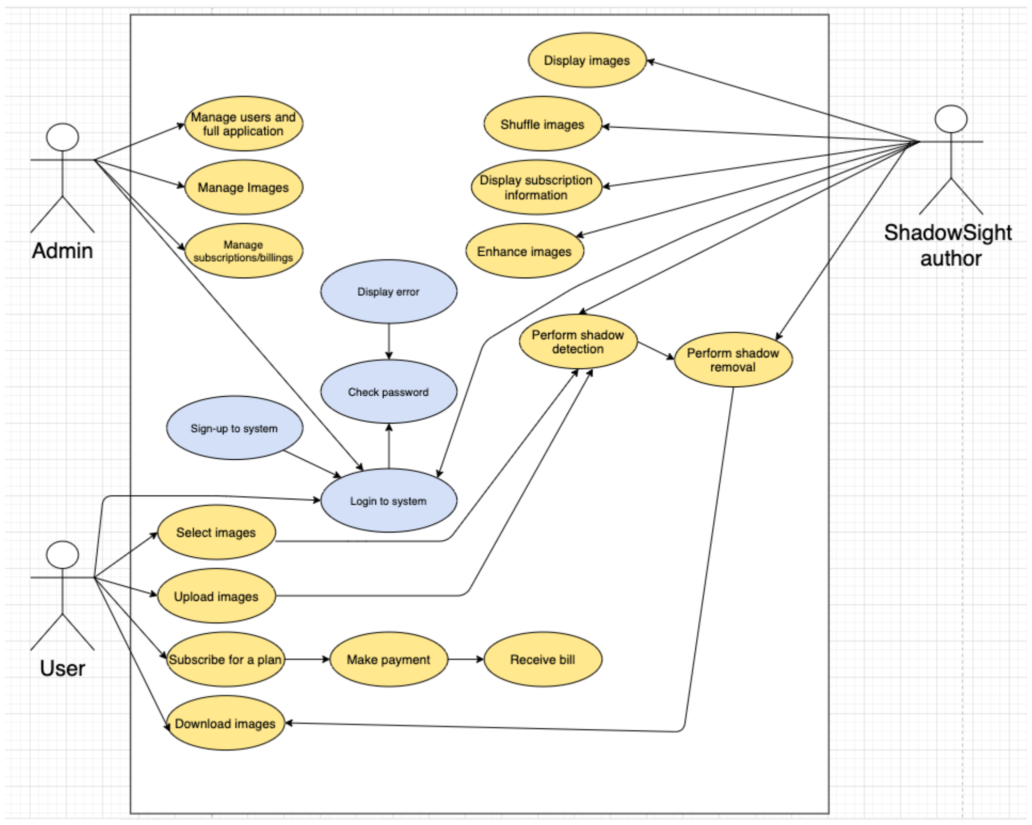

Fig. 1. UML Diagram

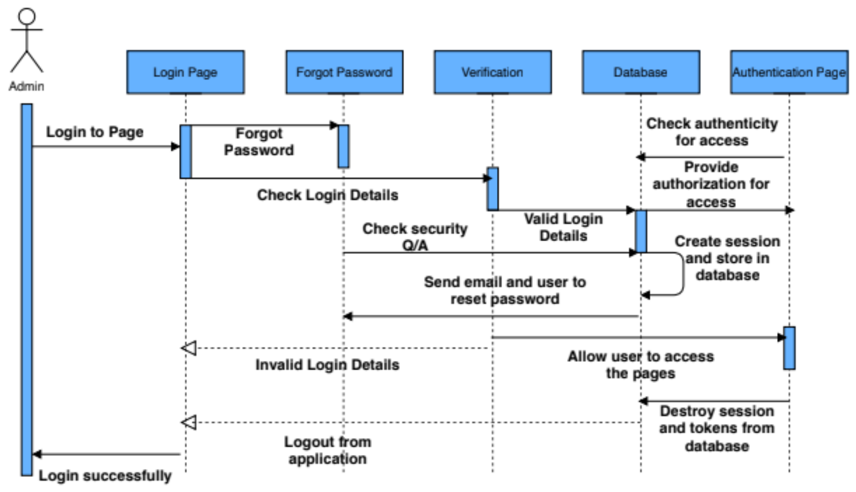

Fig. 2. Sequence Diagram for User Login

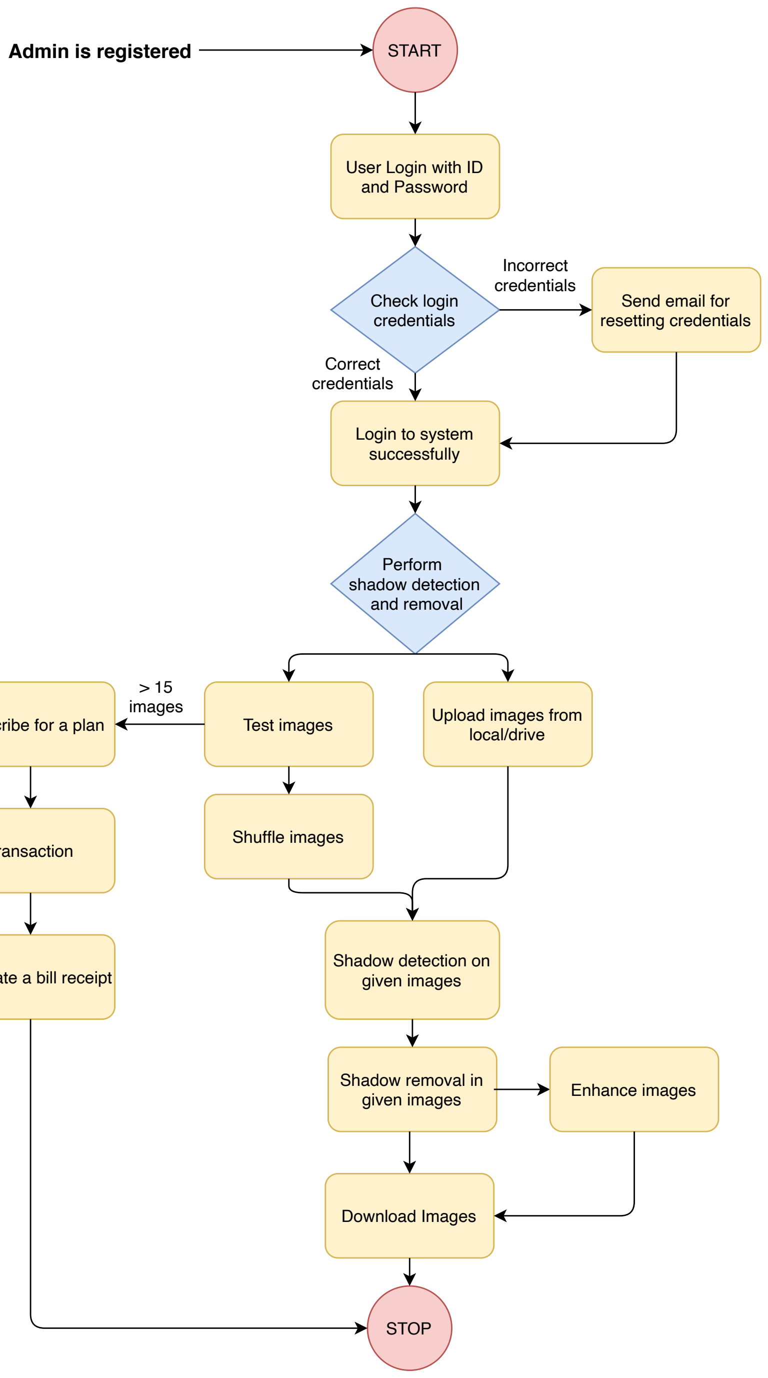
Admin is registered
START
User Login with ID and Password
Check login credentials
Incorrect credentials
Send email for resetting credentials
Correct credentials
Login to system successfully
Perform shadow detection and removal
> 15 images
Subscribe for a plan
Test images
Upload images from local/drive
Transaction
Shuffle images
Generate a bill receipt
Shadow detection on given images
Shadow removal in given images
Enhance images
Download Images
STOP

Fig. 3. Flowchart of ShadowSight Work Flow

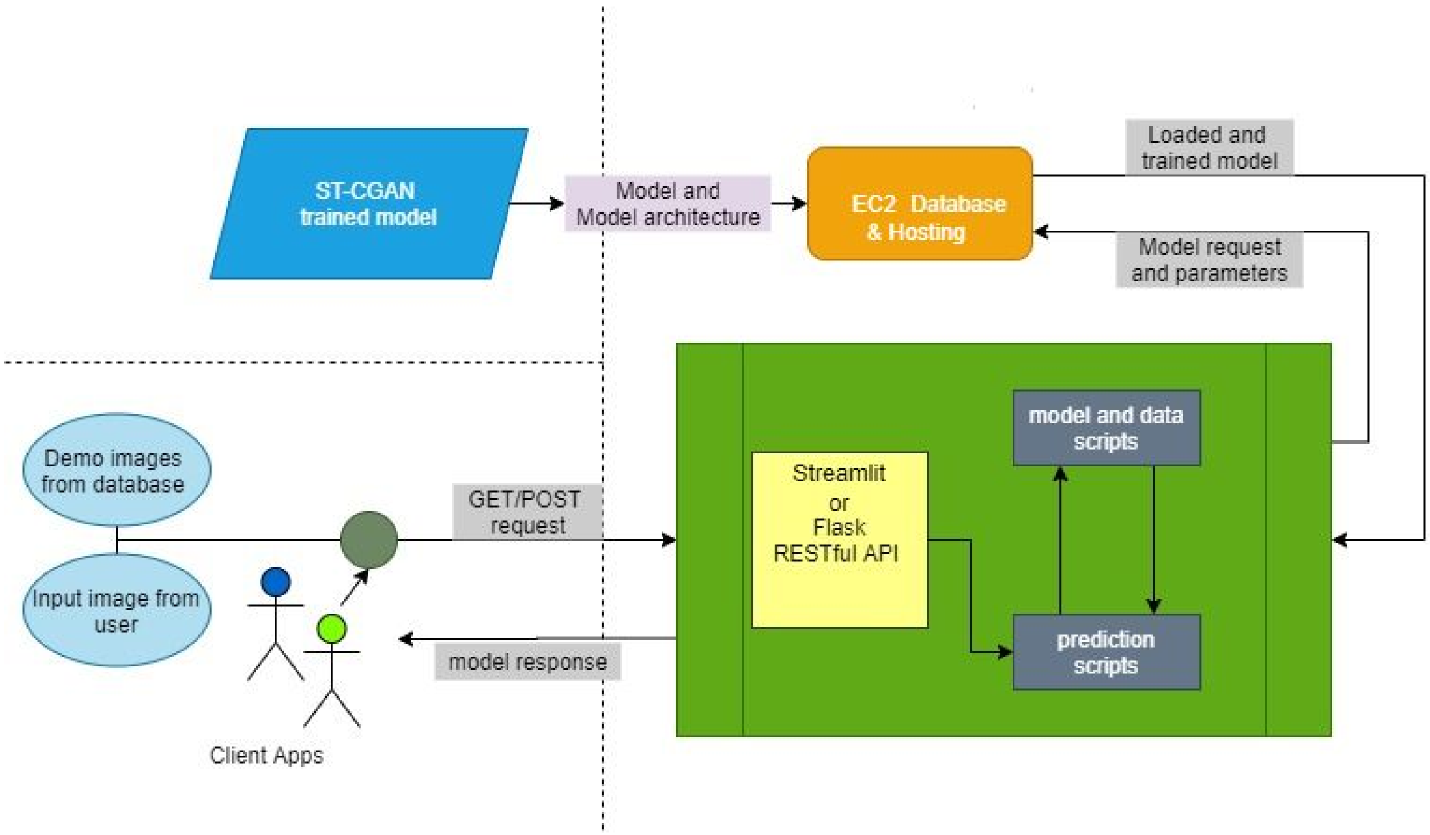

Fig. 4. System Architecture

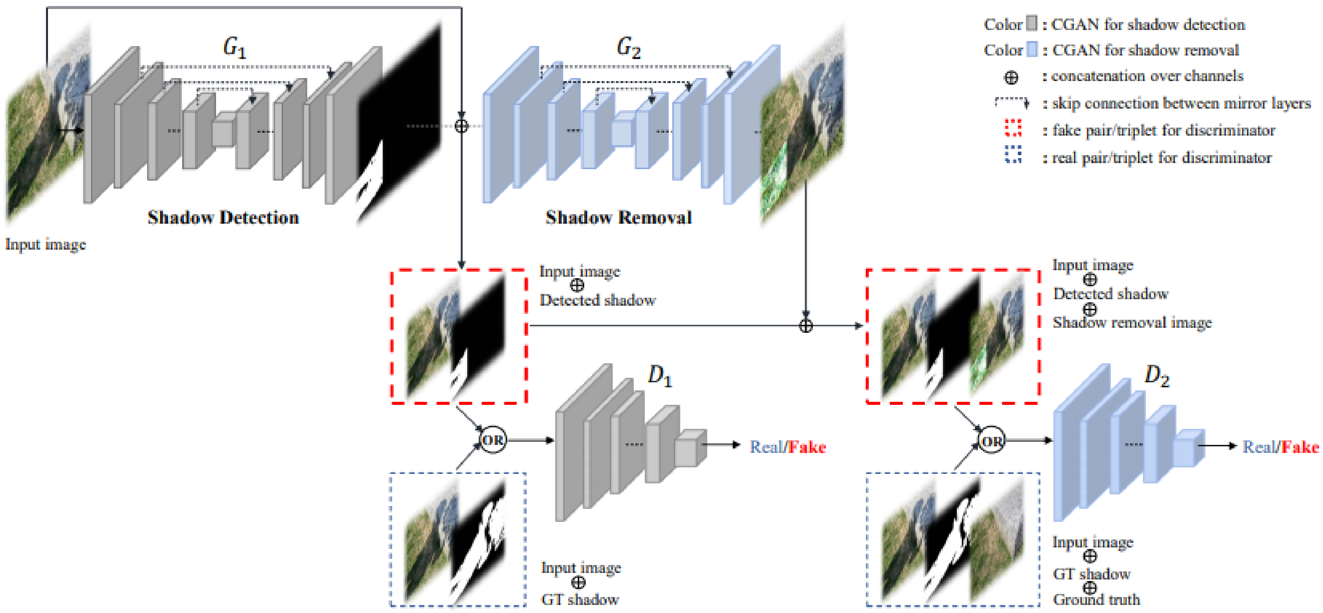
G_1
G_2
Shadow Detection
Shadow Removal
Input image
Color ▨ : CGAN for shadow detection
Color ▨ : CGAN for shadow removal
⊕ : concatenation over channels
: skip connection between mirror layers
: fake pair/triplet for discriminator
: real pair/triplet for discriminator
Input image ⊕ Detected shadow
Input image ⊕ Detected shadow ⊕ Shadow removal image
D_1
D_2
OR
Real/Fake
Input image ⊕ GT shadow
Input image ⊕ GT shadow ⊕ Ground truth

Fig. 5. ST-CGAN Model Architecture

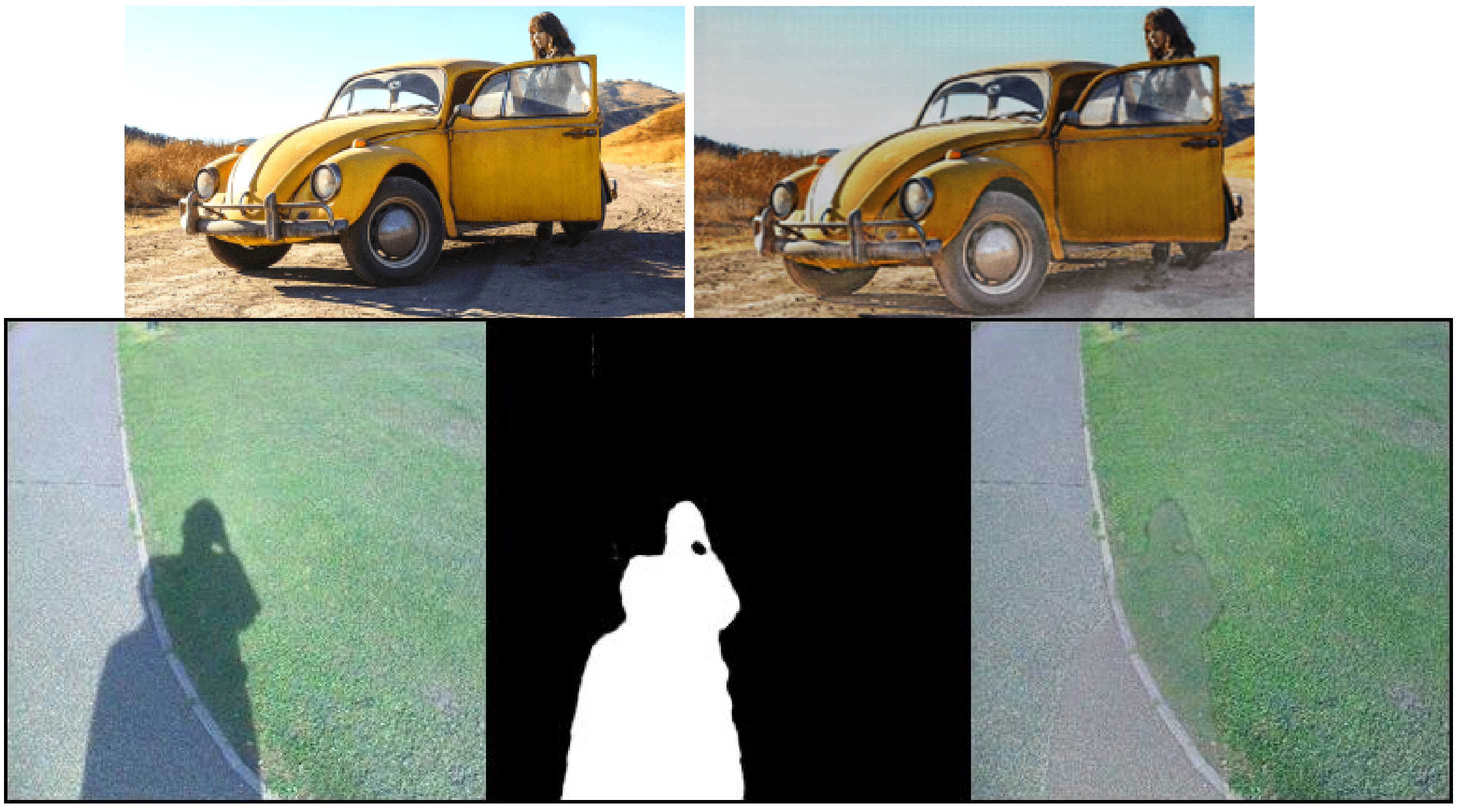
Fig. 6. Reproduced results from ST-CGAN framework[6]

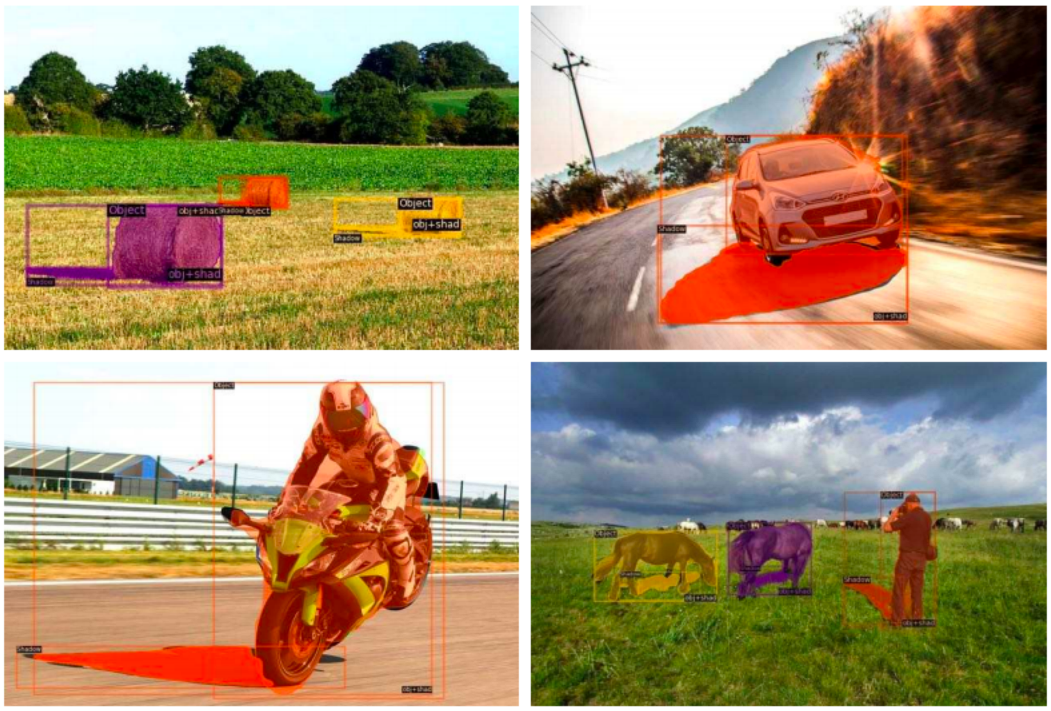

Fig. 7. Reproduced results from LISA framework[5]