A Mini- Project Report

on

“Copy and Store Objects from Surroundings”

Submitted to the

Pune Institute of Computer Technology, Pune

In partial fulfillment for the award of the Degree of

Bachelor of Engineering

in

Information Technology

by

Shyamal Khachane	43225
Manvi Pandya	43233

Under the guidance of

Prof. R. R. Chhajed

Department Of Information Technology

Pune Institute of Computer Technology College of Engineering

Sr. No 27, Pune-Satara Road, Dhankawadi, Pune - 411 043.

2020-2021

CERTIFICATE

This is to certify that the project report entitled

TITLE OF THE PROJECT

Submitted by

Shyamal Khachane	43225
Manvi Pandya	43233

is a bonafide work carried out by them under the supervision of Prof. R. R.Chhajed and it is approved for the partial fulfillment of the requirement of **Computer Laboratory -X** for the award of the Degree of Bachelor of Engineering (Information Technology)

Place	Pune
Date	28/05/2021

ACKNOWLEDGEMENT

We thank everyone who have helped and provided valuable suggestions for successfully creating a wonderful project.

We are very grateful to our guide, Prof. R. R. Chhajed, Head of Department Dr. A. M. Bagade and our principal Dr. P. T. Kulkarni. They have been very supportive and have ensured that all facilities remained available for smooth progress of the project.

We would like to thank our professor and Prof. R. R. Chhajed for providing very valuable and timely suggestions and help.

Shyamal Khachane

Manvi Pandya

ABSTRACT

The drag and drop feature has become second nature to all of us. It is a feature that saves minutes of tedious browsing through the pop-up window, which does not let us switch to a different screen until the file is selected, or the user decides to quit. But an extension of this feature is to drag and store real-world objects into your digital world. Augmented Reality along with Machine Learning and Scripting allows us to copy real-world items into your digital screens just by few clicks. The Project incorporates multiple domains of Computer Science such as Augmented Reality, Networking, Image Processing and Software Development. Also we have limited the scope of our project to storing images to a particular file only, but it can be extended to various softwares like photoshop , microsoft presentation , word , notepad etc in future.

LIST OF FIGURES

Figure Number	Figure Title	Page Number
1	Architecture of BASNet	8
2	Connection of phone and target machine using expo	9
3	Connection of BASNet using endpoint	9
4	Connection to BASNet code	10
5	Connection to BASNet code	10
6	Capture image	11
7	Processing the image - Removal of background	11
8	Log of preprocessing	12
9	Log of preprocessing	12

LIST OF TOPICS

Description	Page number
1. INTRODUCTION	7
2. SCOPE AND OBJECTIVE	7
3. SYSTEM ARCHITECTURE/PROJECT FLOW	8
4. CODE AND SNAPSHOT	10
5. RESULT	11
6. CONCLUSION AND FUTURE SCOPE	13
7. REFERENCE	13

INTRODUCTION

Augmented Reality has enhanced our real-world experience with various features both within and outside the digital world. The process of pasting real-world objects by clicking the snap of an object, eliminating the background, and sharing it with the desktop is tedious. With Augmented Reality, it replaces this whole process by pointing your camera selecting the desired object, making it efficient by bypassing snapping, masking and saving thus making it very efficient for the users. With the combination of AR, ML and user interface simplification techniques coming together this can be achieved. A community provided endpoint is used - by launching a local server with an endpoint.

SCOPE & OBJECTIVES

The scope of this project is limited to only storing the captured object automatically in the user's personal computer. After the user clicks a picture of the object with its surroundings, the background will be erased and the object will be extracted out of the image and stored to the desired machine.

The objectives include :

1. Connecting user's phone and personal computer to the same network to initiate data transfer.

2. Processing the image (mainly using BASNet)

3. Successfully storing it onto the target machine

SYSTEM ARCHITECTURE & PROJECT FLOW

Major sections of the project are:

1. Capturing the image
2. Processing the image to remove the background (extracting the object)
3. Saving it to the target machine

BASNet architecture enables the identification of the real-world object and masks it out from its surrounding.BASNet focuses and aims at accurately segmenting the pixels of prominent objects in an input image. BASNet achieves accurate salient object segmentation with high-quality boundaries.

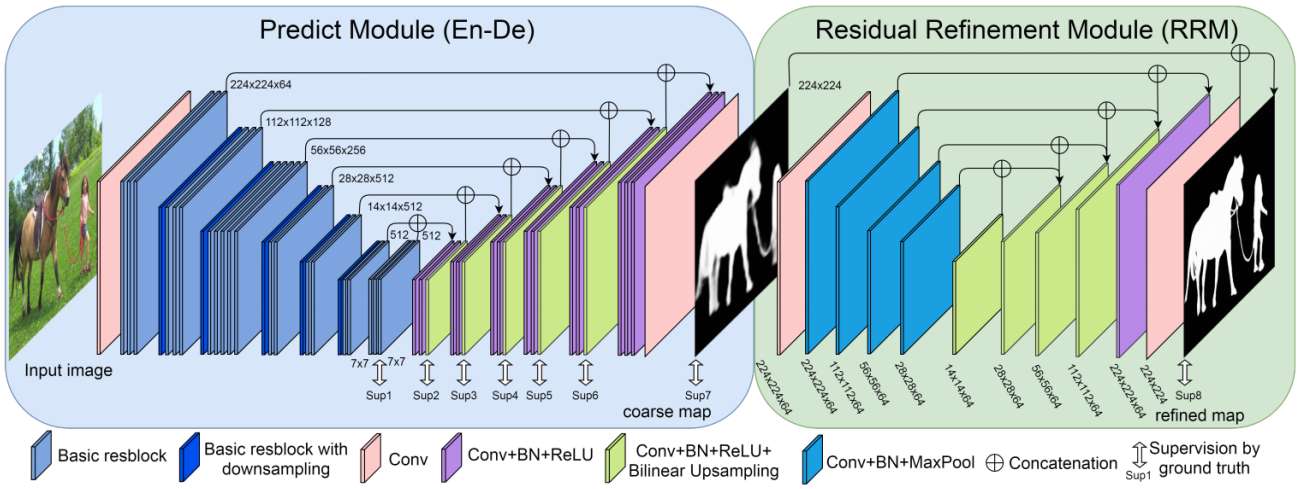

Fig 1 : Architecture of BASNet

The app will incorporate 3 independent modules:
1. The mobile application

2. The local server

The interface between the mobile app and Desktop.

3. The object detection/background removal service

This service will be used in the mobile application which can reduce the noise and help in proper segmentation and detection of actual objects from the camera image.

Connection of phone and target machine :

Both these devices are connected to the same network. Expo CLI is a command line app used as an interface between them. The web based GUI of expo creates a QR code, which is then scanned by the expo go app (Android) or expo client (iOS) for establishing a connection.

Figure 2 : Connection of phone and target machine using expo

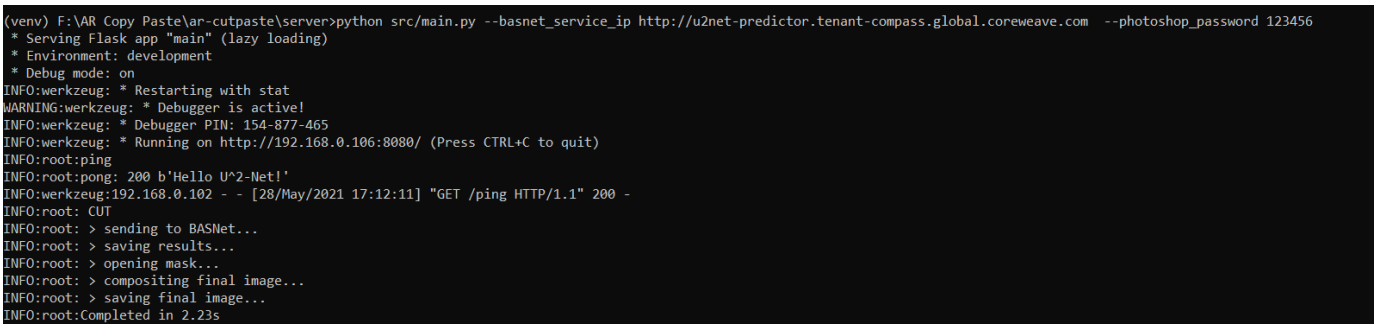

Figure 3 : Connection to BASNet using endpoint

CODE & SNAPSHOT

```
res = requests.post(args.basnet_service_ip, headers=headers, files=files )
# logging.info(res.status_code)

# Save mask locally.
logging.info(' > saving results...')
with open('cut_mask.png', 'wb') as f:
    f.write(res.content)
    # shutil.copyfileobj(res.raw, f)

logging.info(' > opening mask...')
mask = Image.open('cut_mask.png').convert("L").resize((256,256),resample=Image.BICUBIC, reducing_gap=2.0)

# Convert string data to PIL Image.
logging.info(' > compositing final image...')
ref = Image.open(io.BytesIO(data))
empty = Image.new("RGBA", ref.size, 0)
img = Image.composite(ref, empty, mask)

# TODO: currently hack to manually scale up the images. Ideally this would
# be done respective to the view distance from the screen.
img_scaled = img.resize((img.size[0] * 3, img.size[1] * 3))

# Save locally.
logging.info(' > saving final image...')
img_scaled.save('cut_current.png')

# Save to buffer
buff = io.BytesIO()
img.save(buff, 'PNG')
buff.seek(0)

# Print stats
logging.info(f'Completed in {time.time() - start:.2f}s')

# Return data
return send_file(buff, mimetype='image/png')
```

Figure 4 : Connection to BASNet code

```
# Ping to wake up the BASNet service.
@app.route('/ping', methods=['GET'])
def ping():
    logging.info('ping')
    r = requests.get(args.basnet_service_ip, headers={'Host': args.basnet_service_host})
    logging.info(f'pong: {r.status_code} {r.content}')
    return 'pong'


# The cut endpoints performs the salience detection / background removal.
# And store a copy of the result to be pasted later.
@app.route('/cut', methods=['POST'])
def save():
    start = time.time()
    logging.info(' CUT')

    # Convert string of image data to uint8.
    if 'data' not in request.files:
        return jsonify({
            'status': 'error',
            'error': 'missing file param `data`'
        }), 400
    data = request.files['data'].read()
    if len(data) == 0:
        return jsonify({'status:': 'error', 'error': 'empty image'}), 400

    # Save debug locally.
    with open('cut_received.jpg', 'wb') as f:
        f.write(data)

    # Send to BASNet service.
    logging.info(' > sending to BASNet...')
    headers = {}
    if args.basnet_service_host is not None:
        headers['Host'] = args.basnet_service_host
    files= {'data': open('cut_received.jpg', 'rb')}
```

Figure 5 : Connection to BASNet code

RESULT

The image was preprocessed and stored within 5.8 seconds.

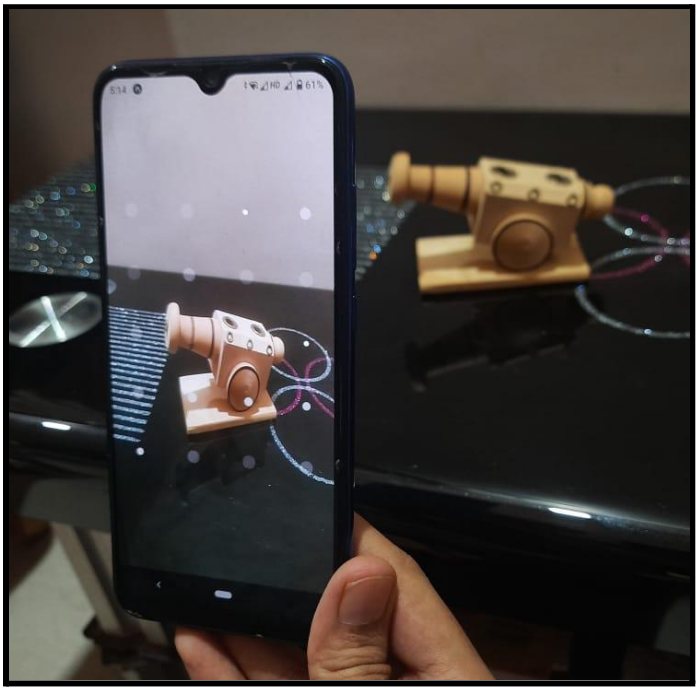

Figure 6: Capturing the image

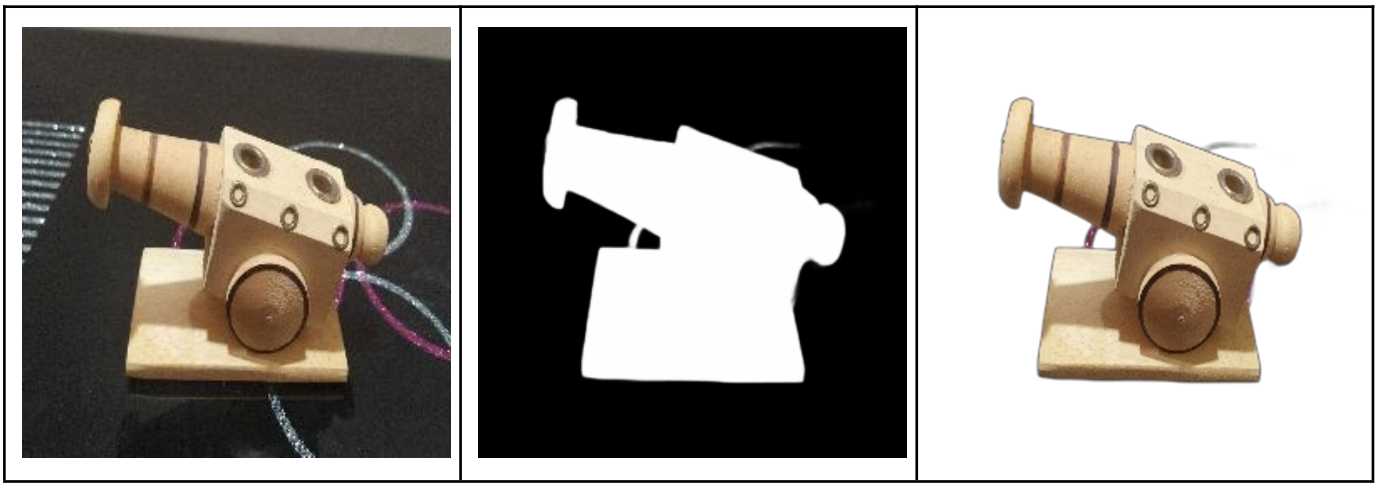

Figure 7 : Preprocessing
Figure 7.1 : Cut Received
Figure 7.2 : Cut Mask
Figure 7.3 : Cut Current

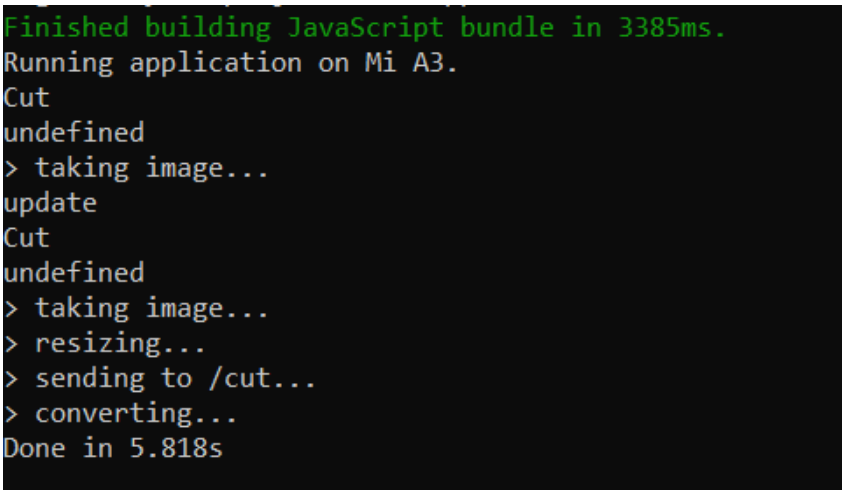

Figure 8 : Log of preprocessing

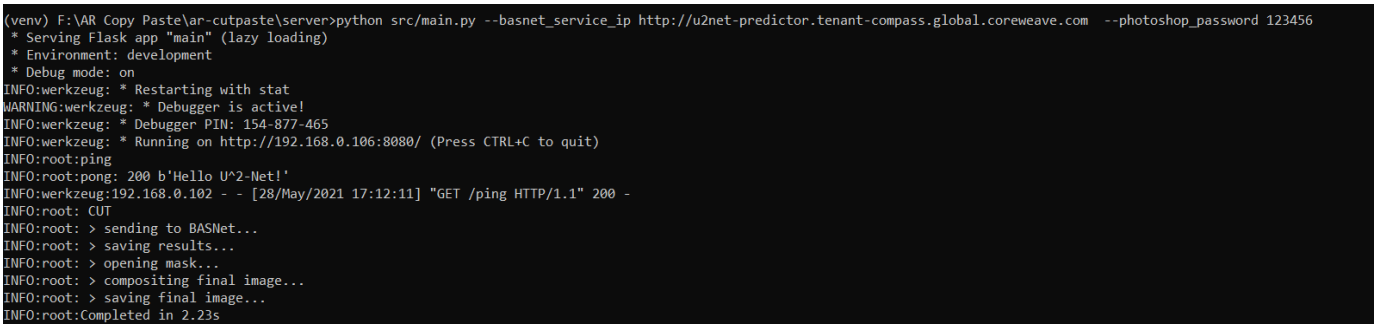

Figure 9 : Log of preprocessing

Method	Backbone	Training data		SOD [45]			ECSSD [68]			DUT-OMRON [69]			PASCAL-S [37]			HKU-IS [33]			DUTS-TE [62]		
		Train	#Images	$maxF_\beta$	$relaxF_\beta^b$	MAE	$maxF_\beta$	$relaxF_\beta^b$	MAE	$maxF_\beta$	$relaxF_\beta^b$	MAE	$maxF_\beta$	$relaxF_\beta^b$	MAE	$maxF_\beta$	$relaxF_\beta^b$	MAE	$maxF_\beta$	$relaxF_\beta^b$	MAE
Ours	ResNet-34	DT	10553	0.851	0.603	0.114	0.942	0.826	0.037	0.805	0.694	0.056	0.854	0.660	0.076	0.928	0.807	0.032	0.860	0.758	0.047
PiCANetR [39]	ResNet-50	DT	10553	0.856	0.528	0.104	0.935	0.775	0.046	0.803	0.632	0.065	0.857	0.598	0.076	0.918	0.765	0.043	0.860	0.696	0.050
BMPM [72]	VGG-16	DT	10553	0.856	0.562	0.108	0.928	0.770	0.045	0.774	0.612	0.064	0.850	0.617	0.074	0.921	0.773	0.039	0.852	0.699	0.048
R^3Net+ [6]	ResNeXt	MK	10000	0.850	0.431	0.125	0.934	0.759	0.040	0.795	0.599	0.063	0.834	0.538	0.092	0.915	0.740	0.036	0.828	0.601	0.058
PAGRN [76]	VGG-19	DT	10553	-	-	-	0.927	0.747	0.061	0.771	0.582	0.071	0.847	0.594	0.0895	0.918	0.762	0.048	0.854	0.692	0.055
RADF+ [19]	VGG-16	MK	10000	0.838	0.476	0.126	0.923	0.720	0.049	0.791	0.579	0.061	0.830	0.515	0.097	0.914	0.725	0.039	0.821	0.608	0.061
DGRL [65]	ResNet-50	DT	10553	0.848	0.502	0.106	0.925	0.753	0.042	0.779	0.584	0.063	0.848	0.569	0.074	0.913	0.744	0.037	0.834	0.656	0.051
RAS [4]	VGG-16	MB	2500	0.851	0.544	0.124	0.921	0.741	0.056	0.786	0.615	0.062	0.829	0.560	0.101	0.913	0.748	0.045	0.831	0.656	0.059
C2S [36]	VGG-16	M30K	30000	0.823	0.457	0.124	0.910	0.708	0.055	0.758	0.565	0.072	0.840	0.543	0.082	0.896	0.717	0.048	0.807	0.607	0.062
LFR [73]	VGG-16	MK	10000	0.828	0.479	0.123	0.911	0.694	0.052	0.740	0.508	0.103	0.801	0.499	0.107	0.911	0.731	0.040	0.778	0.556	0.083
DSS+ [17]	VGG-16	MB	2500	0.846	0.444	0.124	0.921	0.696	0.052	0.781	0.559	0.063	0.831	0.499	0.093	0.916	0.706	0.040	0.825	0.606	0.056
NLDF+ [41]	VGG-16	MB	2500	0.841	0.475	0.125	0.905	0.666	0.063	0.753	0.514	0.080	0.822	0.495	0.098	0.902	0.694	0.048	0.813	0.591	0.065
SRM [64]	ResNet-50	DT	10553	0.843	0.392	0.128	0.917	0.672	0.054	0.769	0.523	0.069	0.838	0.509	0.084	0.906	0.680	0.046	0.826	0.592	0.058
Amulet [74]	VGG-16	MK	10000	0.798	0.454	0.144	0.915	0.711	0.059	0.743	0.528	0.098	0.828	0.541	0.100	0.897	0.716	0.051	0.778	0.568	0.084
UCF [75]	VGG-16	MK	10000	0.808	0.471	0.148	0.903	0.669	0.069	0.730	0.480	0.120	0.814	0.493	0.115	0.888	0.679	0.062	0.773	0.518	0.112
MDF [35]	R-CNN [9]	MB	2500	0.746	0.311	0.192	0.832	0.472	0.105	0.694	0.406	0.092	0.759	0.343	0.142	0.860	0.594	0.129	0.729	0.447	0.099

CONCLUSION & FUTURE SCOPE

Thus, we were able to capture, resize the image and send it to BASNet endpoint for further processing mainly, background removal. Thereafter, the received image, mask and final object image was stored in the target machine.

Scope of the project can be extended by allowing the image to be pasted on various softwares like photoshop, word, presentation, etc. It will certainly add value to the creation of business presentations or to work on an image editing tool.

REFERENCES

1. https://openaccess.thecvf.com/content_CVPR_2019/papers/Qin_BASNet_Boundary-Aware_Salient_Object_Detection_CVPR_2019_paper.pdf
2. https://github.com/NathanUA/BASNet

3. https://github.com/cyrildiagne/ar-cutpaste