Assignment 3

Robot Vision practical

Friedrich Fraundorfer (fraundorfer@icg.tugraz.at)

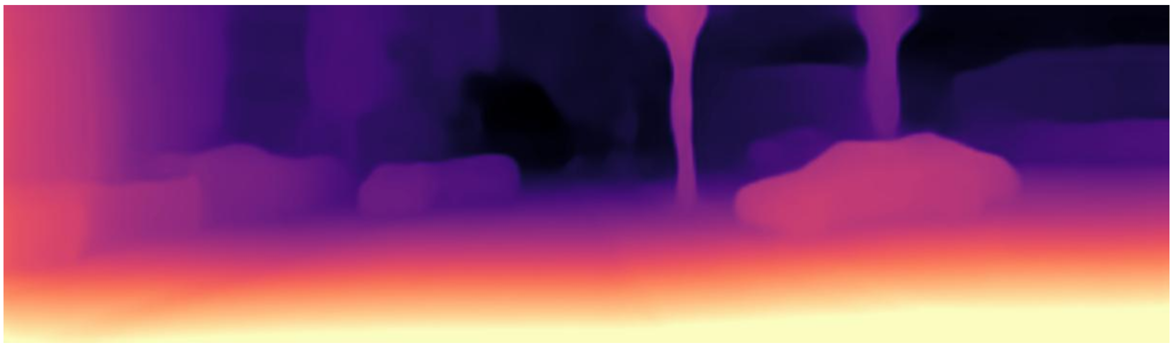
Submission: Submit the completed assignment (code, output data and report) using the TeachCenter submission system as a single ZIP File. Documents need to be submitted using the PDF format.
Use the following naming scheme for the main source files: “group??_assgn3_task?.cpp”.
Use the following naming scheme for the overall zip package: “group??_assgn3.zip”.

Deadline: June 15, 2020, 23:55h
Questions: If you have any questions about the exercises send a mail to chetan.kumar@student.tugraz.at.

Task 1: Predict depth of given images with a Deep Network (15pts)

The goal of this task is to use a monocular depth prediction network to predict a depth image for the given set of images. We will be using the monodepth2 network, which is a state-of-the-art network for depth prediction as per the KITTI benchmark. - https://github.com/nianticlabs/monodepth2

KITTI Depth Vision benchmark: http://www.cvlibs.net/datasets/kitti/eval_depth.php?benchmark=depth_prediction



You are to use one of the pretrained models (use pretrained models of type *stereo or mono+stereo* which have a direct scaling factor provided to convert depth to meters) to run inference on the provided directory of 10 randomly chosen KITTI images.

Deliverables: Include your prediction as a .png file, both in the report and separately. In the report, comment on how the network performs with respect to the ground truth, i.e. if it performs badly in a certain region, etc. You should perform this evaluation for the image **2011_09_26_drive_0095_sync_image_0000000194_image_02.png.**

Task 2: Compute error between prediction and ground truth (10 pts)

Using your predictions from the previous task, please compute the error between the prediction and ground truth images with the Root Mean Squared Error(RMSE) function.

You will have to convert your prediction into the metric scale. This scaling factor is provided in the evaluation script of monodepth2 for the pretrained networks (evaluate_depth.py, which also contains functions useful to evaluate RMSE and several other types of errors).

Deliverables: Report the RMSE between the predicted depth maps and the ground truths for each of the 10 images provided.

Other information:

It is advised to use Google Colab to prototype your chosen networks. Colab provides a GPU enabled environment preinstalled with the latest deep learning libraries (TensorFlow, PyTorch, etc.). Colab can be used as an enhanced Jupyter Notebook with the latest ML libraries and CUDA toolkits pre-installed – indeed, Colab runs .ipynb files.

In general, to run any Neural Network implementation on Colab, please take a look at the instructions that comes with the repository. Unless there is a specific older version of some library required by the implementation (eg. TensorFlow < 2.0, PyTorch < 1.0, some old CUDA version, etc.), you can usually go straight ahead and clone the repository in Colab and try to run inference on the input as directed by the repository's README.

Here is a video tutorial on the basics of Colab:

https://www.youtube.com/watch?v=vVe648dJOdI&t=625s

Here is a link to a GitHub repository that points to several Colab Notebooks with ready to use implementations of many state of the art solutions to Computer Vision tasks.

https://github.com/tugstugi/dl-colab-notebooks