

## SOURCES

### SCRIPTS

Github : <https://github.com/pat92fr/Carre92/tree/master/02%20-%20Logiciel%20PC/03%20-%203DCNN%20Pipeline>






















### DATASET SOURCE

Onedrive : <https://1drv.ms/u/s!AgPL1p8-kSaRg9JQD1uURRMgiD2Low?e=h9OGPs>

Télécharger et décompresser les fichiers dataset\_00n.rar

### INSTALLATION

Le répertoire de travail doit contenir les éléments suivants :

	dataset_001	25/08/2019 23:27	Dossier de fichiers	
	dataset_002	03/09/2019 23:28	Dossier de fichiers	
	dataset_003	06/09/2019 20:12	Dossier de fichiers	
	dataset_004	06/09/2019 20:12	Dossier de fichiers	
	dataset_train_valid_3D	07/09/2019 09:49	Dossier de fichiers	
	model	04/09/2019 01:46	Dossier de fichiers	
	testset	04/09/2019 20:54	Dossier de fichiers	
	video	07/09/2019 08:43	Dossier de fichiers	
	00_Labels.py	25/08/2019 23:07	Python File	7 Ko
	01_Preview_Dataset.py	07/09/2019 08:33	Python File	1 Ko
	02_Dataset_2_Video.py	07/09/2019 01:44	Python File	2 Ko
	03_Build_3D_Dataset.py	07/09/2019 09:48	Python File	4 Ko
	04_Train.py	07/09/2019 02:37	Python File	8 Ko
	05_Predict.py	05/09/2019 23:04	Python File	10 Ko
	my_constants.py	07/09/2019 01:57	Python File	1 Ko
	my_datasettools.py	07/09/2019 02:40	Python File	4 Ko
	my_modeltools.py	07/09/2019 03:09	Python File	3 Ko
	my_parameters.py	07/09/2019 02:20	Python File	4 Ko
	notice.docx	07/09/2019 10:21	Document Micros...	148 Ko
	README.txt	07/09/2019 02:24	Document texte	1 Ko
	TODO.txt	07/09/2019 02:47	Document texte	1 Ko

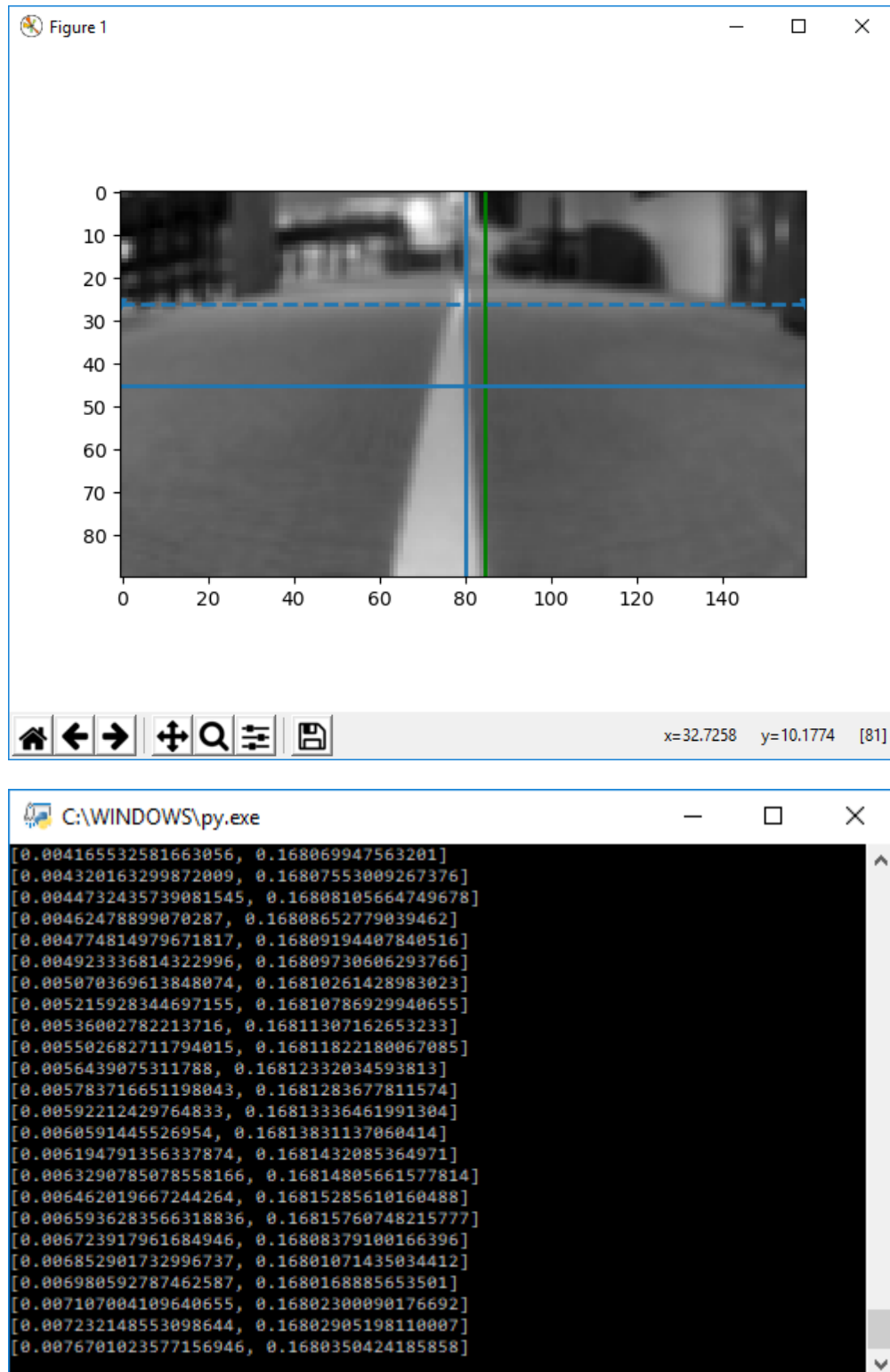
### FICHIER DE CONFIGURATION

Le fichier « **my\_parameters.py** » contient toutes les données de configuration de toutes les étapes.

## PROCEDURE

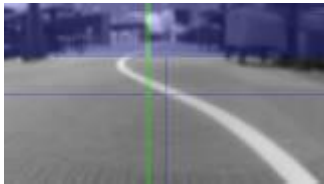
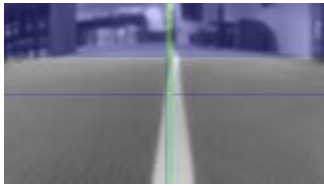
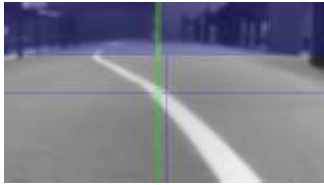
### OUTIL DE VISUALISATION DES DATASET SOURCE (OPTIONNEL)

Lancer « 01\_Preview\_Dataset.py » pour visualiser l'ensemble des DATASET. La sortie standard affiche la valeur des commandes de DIRECTION et de GAZ pour chaque image. La fenêtre graphique affiche l'image prétraitées (filtrage flou + N&B) avec un repère (bleu) et la commande de direction (vert). La ligne pointillée bleue délimite la zone supérieure de l'image, éliminée avant traitement par le CNN.







## CONVERSION DES DATASET SOURCE EN UNE VIDEO (OPTIONNEL)

Lancer « 02\_Dataset\_2\_Video.py » pour convertir l'ensemble des DATASET en un seul média. Le fichier est disponible dans le dossier « vidéo » sous le nom « dataset\_preview.avi ». Le bandeau supérieur bleu correspond à la zone de l'image, éliminée avant traitement par le CNN.



## CREATION DES DATASET 3D TRAIN-VALID (UNE SEULE FOIS)

Lancer « 03\_Build\_3D\_Dataset.py » et patienter ! Les fichiers binaires contenant les séquences d'image (DATASET 3D) sont générés et stockés dans le dossier « dataset\_train\_valid\_3D ».

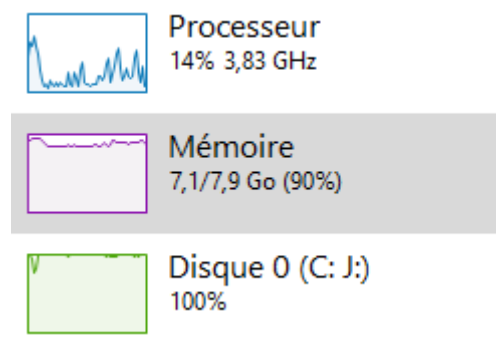
Nom	Modifié le	Type	Taille
 Xtrain.bin.npy	07/09/2019 10:19	Fichier NPY	5 419 997 Ko
 Xvalid.bin.npy	07/09/2019 10:19	Fichier NPY	285 272 Ko
 Ytrain.bin.npy	07/09/2019 10:19	Fichier NPY	1 004 Ko
 Yvalid.bin.npy	07/09/2019 10:19	Fichier NPY	53 Ko

La génération et l'augmentation des DATASET se basent sur les paramètres du fichier « my\_parameters.py », notamment :

```
# picture sequence for CNN
depth = 6          ## default 6 frames processed by CNN
skip = 3           ## default 3 frames skipped between each pair of frames processed by CNN

# hyperparameters
hyp_train_valid_dataset_ratio = 0.05
```

*Prérequis : optimiser/vérifier la configuration du swap. La taille des DATASET en mémoire dépasse 16Go.*



## APPRENTISSAGE DU 3D CNN (REPETER A CHAQUE CHANGEMENT DE PARAMETRES)

Lancer « 04\_Train.py » et patienter ! Au lancement, l'architecture du CNN est affichée.

Layer (type)	Output Shape	Param #
cropping3d_1 (Cropping3D)	(None, 6, 64, 160, 1)	0
lambda_1 (Lambda)	(None, 6, 64, 160, 1)	0
conv3d_1 (Conv3D)	(None, 4, 20, 52, 16)	1216
max_pooling3d_1 (MaxPooling3D)	(None, 4, 10, 26, 16)	0
conv3d_2 (Conv3D)	(None, 2, 8, 24, 32)	13856
max_pooling3d_2 (MaxPooling3D)	(None, 2, 4, 12, 32)	0
flatten_1 (Flatten)	(None, 3072)	0
dense_1 (Dense)	(None, 256)	786688
batch_normalization_1 (Batch Normalization)	(None, 256)	1024
activation_1 (Activation)	(None, 256)	0
dropout_1 (Dropout)	(None, 256)	0
dense_2 (Dense)	(None, 128)	32896
batch_normalization_2 (Batch Normalization)	(None, 128)	512
activation_2 (Activation)	(None, 128)	0
dropout_2 (Dropout)	(None, 128)	0
dense_3 (Dense)	(None, 2)	258
activation_3 (Activation)	(None, 2)	0
Total params: 836,450		
Trainable params: 835,682		
Non-trainable params: 768		

Pendant l'apprentissage, le cout et l'erreur est affichée pour les DATASET d'entraînement et de validation, à chaque fin d'EPOCH.

```
Epoch 1/50
- 124s - loss: 0.2308 - mean_squared_error: 0.1796 - val_loss: 0.0295 - val_mean_squared_error: 0.0291
Epoch 2/50
- 89s - loss: 0.0236 - mean_squared_error: 0.0234 - val_loss: 0.0217 - val_mean_squared_error: 0.0216
Epoch 3/50
- 89s - loss: 0.0202 - mean_squared_error: 0.0201 - val_loss: 0.0181 - val_mean_squared_error: 0.0179
Epoch 4/50
- 96s - loss: 0.0176 - mean_squared_error: 0.0174 - val_loss: 0.0159 - val_mean_squared_error: 0.0157
Epoch 5/50
- 96s - loss: 0.0156 - mean_squared_error: 0.0153 - val_loss: 0.0136 - val_mean_squared_error: 0.0133
Epoch 6/50
- 74s - loss: 0.0136 - mean_squared_error: 0.0132 - val_loss: 0.0115 - val_mean_squared_error: 0.0112
Epoch 7/50
- 75s - loss: 0.0118 - mean_squared_error: 0.0114 - val_loss: 0.0097 - val_mean_squared_error: 0.0093
Epoch 8/50
- 65s - loss: 0.0101 - mean_squared_error: 0.0097 - val_loss: 0.0082 - val_mean_squared_error: 0.0078
```

A la fin de l'apprentissage, le modèle et l'historique sont générés et stockés dans le dossier « model ».

L'apprentissage se base sur les paramètres du fichier « my\_parameters.py », notamment :

```
# CNN parameters
### see build_3d_cnn https://github.com/autorope/donkeycar/blob/dev/donkeycar/parts/keras.py
### Credit: https://github.com/jessecha/DNRRacing/blob/master/3D_CNN_Model/model.py
conv_layers = [
    ('crop3D', picture_height_crop),      ## do not change
    ('norm', 0),                          ## do not change

    ('conv3D', 16, (3,5,5), (1,3,3)),      ## filters, kernel_size, stride
    ('maxpooling3D', (1,2,2), (1,2,2)),    ## size, stride

    ('conv3D', 32, (3,3,3), (1,1,1)),
    ('maxpooling3D', (1,2,2), (1,2,2)),
]
full_connected_hidden_layers= [
    (256, 'relu', 0.1),                  ## default 256 units in 1st hidden layer, 10% dropout
    (128, 'relu', 0.1)                   ## default 128 units in 2nd hidden layer, 10% dropout
]

# hyperparameters
hyp_train_valid_dataset_ratio = 0.05
hyp_batch_size = 128
hyp_epoch = 50
hyp_lr = 0.0001
hyp_lr_decay = 0.0
hyp_l2_regularization = 0.0001
hyp_min_delta=0.0002
hyp_patience=10
```

*Note : Les paramètres “conv\_layers » et « full\_connected\_hidden\_layers” sont à adapter en utilisant les mots clés : « conv3D », « maxpooling3D », « avgpooling3D », « dropout », « batchnorm » tirés de l’API Keras.*

*Note : Le paramètre « epoch » permet de limiter le nombre d’itération d’apprentissage.*

*Note : Le paramètre “lr” correspond au “Learning Rate”*

*Note : Les paramètres « min\_delta » et « patience » sont exploités par la fonction de « Early Stopping ».*

PREDICTION APPLIQUEE AUX DATASET SOURCE (REPETER A CHAQUE APPRENTISSAGE)

## OUTILS DE VERIFICATION

VISUALISATION DE LA DISTRIBUTION DE LA COMMANDE DE DIRECTION

VISUALISATION DU VOLUME EN ENTREE DU 3D CNN