

# Spherical Convolution Neural Networks : Application to cars license plate recognition

Sidney Bakhouché<sup>1</sup>, Quentin Sarda<sup>1</sup>, Thomas Schneider<sup>1</sup> and Sebastien Touche<sup>1</sup>

<sup>1</sup>*Institut Polytechnique des Sciences Avancées, IPSA, Ivry-sur-Seine, France*  
*sidney.bakhouché@ipsa.fr; thomas.schneider@ipsa.fr; quentin.sarda@ipsa.fr; sebastien.touche@ipsa.fr*

**Keywords:** Spherical CNN, YOLO algorithm, object detection, polarization

**Abstract:** In this paper, we propose a solution for object detection on spherical polarized images using a modified version of the YOLO v3 algorithm, an algorithm that uses Convolutional Neural Networks to detect objects in real-time. We adapted it to recognize cars on a data set of pictures with three different polarizations. Our method was able to detect cars on the said polarized images but further development is needed to cast these images on a sphere and see if there is an improvement.

## 1 INTRODUCTION

Object detection is used in many different fields of science and industry for image processing (Boomsma and Frellsen, 2017). The most common type of images that are processed are two dimensional images with a red, blue, and green (RGB) pixel layout. We are interested in a different type of data: spherical polarized images, mostly used in spatial imagery. For example, the pictures taken by Planck, that can photograph polarized lights.

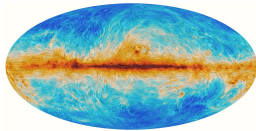


Figure 1: Polarised emission from Milky Way dust pillars by the European Space Agency<sup>2</sup>

There exist many ways to analyze spherical data, with spherical neural networks (Cohen et al., 2018) or by treating the data after projecting it in two dimensions (Boomsma and Frellsen, 2017), resulting in a loss of ratios. In order to improve the lack of existing method to detect objects on polarized spherical data, we adapt the YOLO algorithm (Redmon et al., 2015) so it can recognize objects on polarized spherical data, and possibly, further along the line, embed it on a robot.

<sup>2</sup>[https://www.esa.int/ESA\\_Multimedia/Images/2015/02/Polarised\\_emission\\_from\\_Milky\\_Way\\_dust](https://www.esa.int/ESA_Multimedia/Images/2015/02/Polarised_emission_from_Milky_Way_dust)

First, this document will cover the required information necessary to understand and work with the data used for the algorithm. Then it will make light of different methods already in use to execute spherical convolutions. After this, the architecture of the YOLO v3 algorithm used to do the object detection for this paper will be laid out to understand the approach implemented in this paper. Finally the solution chosen and its implementation will be explained.

## 2 CONTEXT

To begin, we need to define some notions necessary to understand the rest of this paper, such as polarization, object detection, the YOLO algorithm, and three dimensional images.

Light polarization defines the oscillation of the electric field of this electromagnetic wave, which is perpendicular to the direction of propagation. If the oscillation is random, it is called unpolarized, like most natural light in nature. Using a filter, it is possible to isolate the orientation of the field, and therefore set the polarization angle we want out of the filter. This process is used in photography to reduce glare, reflection or reduce the light intake into an objective.

Object detection algorithms are used on image to find specific patterns. Using convolutional neural net-

<sup>4</sup><https://www.baumer.com/ch/de/service-support/know-how/technologische-highlights/polarisation/a/Polarization>

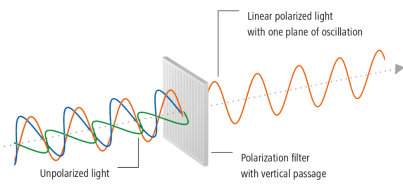


Figure 2: Polarization of light with a filter <sup>4</sup>

work, one feeds data sets to train weights on an algorithm which will then be able to recognize said object from new images. The way it works is that the algorithm has a specific frames in which all pixels have a different weight. It then operates a number of combination on those weighted pixels until it gets a simple output. Then, by comparing the result inside of the frame with the result expected for the object to be detected, the algorithm can tell or not if said object is in that frame. Finally, the frame moves to a different position and tries again until the whole picture has been analysed.

Some examples of those algorithms are:

- **Faster R-CNN (Girshick, 2015):** This algorithm predicts the possible position and size of frames to be used on the picture to optimize the number of iterations it has to do. Indeed, instead of doing the whole picture many times to find where it is and what size it is, it looks right into where the object should be with the right size of frame.
- **Single Shot MultiBox Detector (Liu et al., 2015) :** This algorithm only uses a single deep neural network that adjusts itself as it trains. It has better results than Faster R-CNN on smaller images.

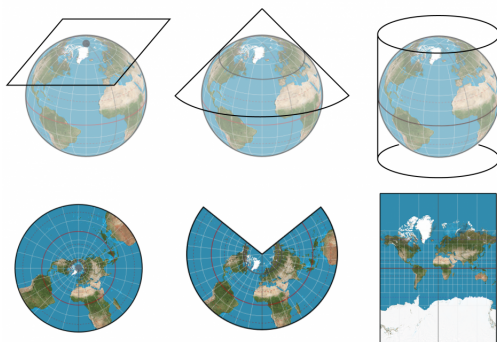


Figure 3: Projection of a sphere on a plane <sup>6</sup>

<sup>6</sup><https://www.cartography.org.uk/wp-content/uploads/2018/01/BM2017-11-20-Session-04-Map-Projections-Giles\Darkes-and-Chris-Wesson.pdf>

In some cases, data can be spread over the surface of a sphere, giving each point of data three points of coordinates instead of two, if it was on a two dimensional plane. This data can be treated in different ways: it can be studied directly on the spherical plane, which then complicates most common mathematical application by forcing them to use the polar coordinates and switches from base  $SO(2)$  to base  $SO(3)$ . An other way to study spherical data is to project it on a two dimensional plane before analyzing it. This process is easier since we already have many ways of working with two dimensional data, but the downside is that the data is deformed. We can either keep angle ratios or distance ratios.

The biggest challenge when working with spherical data is that we do not use the volume of the sphere. This leads to many mathematical properties being unusable when trying to use convolution. Indeed, the surface of the convolutional grid is spread over the sphere and the result of the neural network used on said surface is set on the middle of it. For a spherical surface, the mean of all the points of the surface is not on said surface, but inside the sphere. The solutions

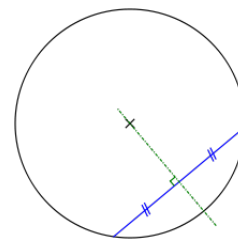


Figure 4: The mean of the points on the surface of the sphere is inside the sphere. <sup>8</sup>

are as stated above, to project the data on a flat surface, or to apply spherical convolutions directly to the spherical data, requiring powerful tools.

### 3 STATE OF THE ART

There exist nowadays papers on algorithms for spherical convolution. We can mostly look at:

- **Convolutions on Spherical Images (Eder and Frahm, 2019) :** In this paper, they use spherical computed data into a CNN for image recognition. The work is focused on the data, as it says : "Rather than changing the tools of convolution, we look to find a better representation of the data". The sphere is approximated as an icosphere (as an "Icosahedral grid") as it provides planes to where

<sup>8</sup>[https://commons.wikimedia.org/wiki/File:Cercle\\_mediatrice\\_corde.svg](https://commons.wikimedia.org/wiki/File:Cercle_mediatrice_corde.svg)

filters can be applied for the CNN training. The results are roughly similar to basics CNNs.

- **SPHERICAL CNNs** (Cohen et al., 2018) : This method uses the Fast Fourier Transform on three dimensional data to bypass projection on a two dimensional plane. Their results have been very conclusive.
- **Learning SO(3) Equivariant Representations with Spherical CNNs** (Esteves et al., 2017) : In this paper a three dimensional object is mapped on a sphere using spherical functions then equiangular spherical sampling is used to apply a Spherical CNN.
- **Grid Based Spherical CNN for Object Detection from Panoramic Images** (Yu and Ji, 2019) : In this paper, they adapt the method to create a SCNN from the two previous papers by applying a grid on the sphere. This help to avoid losing information about the object (in an object detection task).
- **Spherical Fractal Convolutional Neural Networks for Point Cloud Recognition** (Rao et al., 2019) : SCNN adapted for point cloud recognition. This paper showed us that it is possible to adapt a SCNN for object recognition, close to the one in the "Learning SO(3) Equivariant Representations with Spherical CNNs" for different kinds of data (3D data in this case). It also approximate the sphere as an icosphere (segmenting/meshing).

There is currently, very little work about Spherical CNNs. It is a new method that is believed to work better than classical CNNs to train algorithms for object recognition, because the depth and the complex shape of the sphere could be used to add filters and trainable parameters. For 2D data, the current methods (most of them) apply a "rotation equivariant", meaning that the 2D data is computed onto the surface of a sphere, with or without deformations (due from going from a 2D plane to 3D). Some of these studies also apply a mesh on the sphere as it is easier to apply filters for the SCNN. Tests from these studies have shown that SCNNs are performing slightly better than the classical CNNs.

## 4 YOLO V3 NETWORK ARCHITECTURE

"You Only Look Once" is an algorithm that uses a convolutional neural network for object detection. It is one of the fastest object detection algorithms in the field, although not the most accurate, but that makes it ideal for real-time detection tasks,

without loss of too much accuracy. In comparison to recognition algorithms, a detection algorithm does not only predict class labels but also detects locations of objects. The YOLO algorithm divides the image into regions and predicts bounding boxes and probabilities for each region of the image. The architecture of the network is showed below.

	Type	Filters	Size	Output
1x	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
	Convolutional	32	1 × 1	128 × 128
	Convolutional	64	3 × 3	
2x	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
	Convolutional	64	1 × 1	64 × 64
	Convolutional	128	3 × 3	
8x	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
	Convolutional	128	1 × 1	32 × 32
	Convolutional	256	3 × 3	
8x	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
	Convolutional	256	1 × 1	16 × 16
	Convolutional	512	3 × 3	
4x	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
	Convolutional	512	1 × 1	8 × 8
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Figure 5: YOLO V3 architecture <sup>10</sup>

YOLO v3 uses only convolutional layers with an architecture called Darknet-53. It contains 53 convolutional layers, each followed by batch normalization and Leaky ReLU activation. No form of pooling is used, and a convolutional layer with stride 2 is used to downsample the feature maps. This helps in preventing loss of low-level features often attributed to pooling.

In other words, the network downsamples the image by a factor called the stride of the network. For example, if the stride of the network is 32, then an input image of size 416 x 416 will yield an output of size 13 x 13.

YOLO can only detect objects belonging to the classes present in the dataset used to train the network originally. The weights generated, that have been obtained by the YOLO development team when training the network on the COCO dataset, can detect 80 object categories including cars.

<sup>10</sup><https://pjreddie.com/media/files/papers/YOLOv3.pdf>

YOLO v3 makes predictions across 3 different scales. The detection layers are used to make detections of three different sizes, with strides 32, 16, and 8. The network downsamples the input image until the 1st detection layer, where a detection is made using feature maps of a layer with stride 32. Further, layers are upsampled by a factor of 2 and another detection is made with stride 16. The same upsampling procedure is repeated, and a final detection is made at the layer of stride 8. This means, with an input of  $416 \times 416$ , there are detections on scales  $13 \times 13$ ,  $26 \times 26$  and  $52 \times 52$ . The people behind YOLO v3 state that this helps detecting small objects, which was lacking in the previous versions. Upsampling can help the network to learn fine-grained features which are instrumental for detecting small objects.

For an image of size  $416 \times 416$ , YOLO predicts  $((52 \times 52) + (26 \times 26) + 13 \times 13) \times 3 = 10647$  bounding boxes. However, in the case of a car image example, there is only one object, a car. To reduce the detections from 10647 to 1, boxes are filtered on their objectness score. Boxes with scores below a certain threshold (for example below 0.5) are ignored. Next, Non-Maximum Suppression (NMS) intends to fix the problem of multiple detections of the same feature. NMS uses a function called "Intersection over Union", or IoU, that removes the boxes that overlap, so only the best fitted box remains.

In the next part, we explain how we adapted the YOLO v3 algorithm to our dataset.

## 5 DEVELOPED APPROACH

The approach we decided to go with is to modify the YOLO v3 algorithm so it can be used on our data after projecting it in two dimensions. We first analysed the data which consisted of pictures of cars with subsets of polarization. For each picture there was a  $0^\circ$ ,  $45^\circ$  and  $90^\circ$  polarization version of said picture (figure 6).

The images had labels attached so we could train our algorithm to recognize cars.

We trained the YOLO v3 algorithm to recognize cars as displayed in our data set based on the red, blue and green pixels. With each set of pictures was a file which contained the coordinates for the delimitation boxes to detect the cars as well as the stokes parameters that describe the polarization of the images (figure 7).

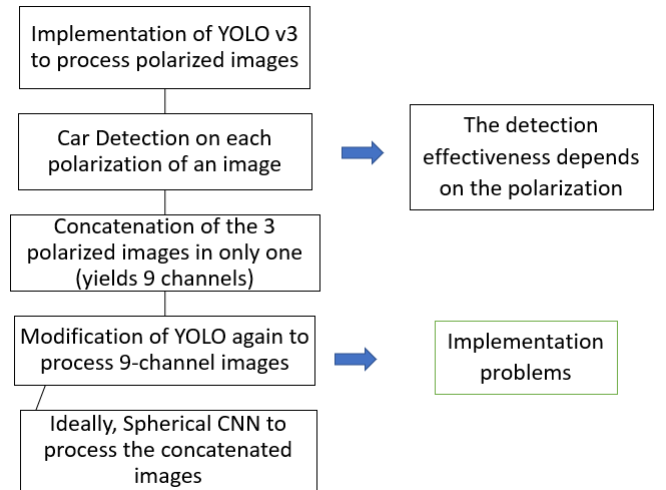


Figure 6: YOLO for polarized spherical data



Figure 7:  $0^\circ$ ,  $45^\circ$  and  $90^\circ$  respectively



Figure 8: Results for  $0^\circ$ ,  $45^\circ$  and  $90^\circ$  respectively after YOLO object detection

Once we knew the algorithm could recognise cars in our data set, we piled up the picture in order to have only one picture with the same height and length but with a depth of nine (the three colors RGB with one polarization for each).

As can be seen in figure 8, the polarization of the image can sometimes change the result of the YOLO algorithm. For that reason, we want to concatenate the images so the result will be homogeneous no matter the polarization.



Figure 9: Different results for  $0^\circ$ ,  $45^\circ$  and  $90^\circ$  respectively after YOLO object detection

## 6 EXPERIMENTAL RESULTS

The data we worked with were pictures with the format  $416 \times 416 \times 3$  (height x length x Red, blue,

green). Each picture was in triple, one for each polarization ( $0^\circ$ ,  $45^\circ$ ,  $90^\circ$ ).

As can be expected, the YOLO algorithm could find the cars with ease on the pictures with the RGB channels. To work on the polarization, we started by concatenating the pictures such that the final format was  $416 \times 416 \times 9$ , the point was to sum up the same picture with its 3 different polarization. That changed our number of channels from 3 to 9, but the YOLO algorithm could not take 9 channels as an input parameters, the maximum size is 3. We could not get any results as we could not manage to adjust the YOLO algorithm.

So we decided to act differently on our data set in order to keep the theoretical 9 channels but keeping the size of 3 needed for the YOLO algorithm. We then reshaped our data, by projecting the 9 channels, as  $416 \times 1248 \times 3$  (or :  $416 \times [416 \times 3] \times 3$ ). The problem in this case was that the YOLO algorithm could not compute this kind of shape. Our biggest issue with reshaping data that had not the same height and length was that all the training done previously on the data set was outdated because of the new input size.

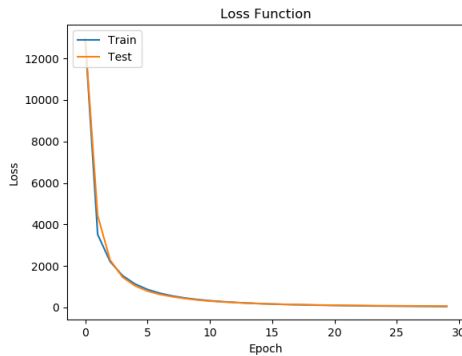


Figure 10: Loss function of the algorithm on the modified data

Finally, we implemented the algorithm so it can take  $832 \times 832 \times 3$  images as inputs. We did as such so that the images can be computed by the algorithm without losing the polarimetric information from the  $416 \times 1248 \times 3$  format we previously concatenated. This simplification is working, but the results are not as good as the tests with the unprocessed data set ( $416 \times 416 \times 3$ ). But since the processed  $416 \times 416 \times 3$  doesn't have the polarimetric information, we have to use the  $832 \times 832 \times 3$  because YOLO only accepts square data, and since the bounding boxes are drawn after the data has been resized, and the detection boxes depend on the size of the object, we lose some information with the format  $832 \times 832 \times 3$ , and end up sometimes with some boxes not being closed.

## 7 CONCLUSIONS

The outcome of our work was conclusive, as we were able to adapt YOLO to our needs. From here on out, the next step would be to use the algorithm on a larger scale data set and with a higher quality of images. The different solutions we can think of in order to improve the algorithm are:

- Changing the core code of YOLO in order for it to treat our input data with 9 channels. The main program already exists, which would save a lot of time, but modifying an already existing code can lead to many incompatibility problems if done incorrectly, which was our biggest obstacle. We would require to change the data we input for every new different set.
- Creating a new algorithm from scratches with the same architecture as YOLO. This would be the longest solution, but would ensure the result is adapted to our needs, and could even be optimized as to not be as heavy as the existing YOLO.

As for embedding the algorithm, it has been done for real time object detection (Rosebrock, 2020), so it could be done with either YOLO v3 or the Tiny module, which is faster than the actual YOLO v3.

## REFERENCES

- Boomsma, W. and Frellsen, J. (2017). Spherical convolutions and their application in molecular modelling. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30*, pages 3433–3443. Curran Associates, Inc.
- Cohen, T. S., Geiger, M., Köhler, J., and Welling, M. (2018). Spherical cnns. *CoRR*, abs/1801.10130.
- Eder, M. and Frahm, J. (2019). Convolutions on spherical images. *CoRR*, abs/1905.08409.
- Esteves, C., Allen-Blanchette, C., Makadia, A., and Danilidis, K. (2017). 3d object classification and retrieval with spherical cnns. *CoRR*, abs/1711.06721.
- Girshick, R. (2015). Fast r-cnn. In *The IEEE International Conference on Computer Vision (ICCV)*.
- Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S. E., Fu, C., and Berg, A. C. (2015). SSD: single shot multi-box detector. *CoRR*, abs/1512.02325.
- Rao, Y., Lu, J., and Zhou, J. (2019). Spherical fractal convolutional neural networks for point cloud recognition. pages 452–460.
- Redmon, J., Divvala, S. K., Girshick, R. B., and Farhadi, A. (2015). You only look once: Unified, real-time object detection. *CoRR*, abs/1506.02640.
- Rosebrock, A. (2020). Yolo and tiny-yolo object detection on the raspberry pi and movidius ncs.
- Yu, D. and Ji, S. (2019). Grid based spherical cnn for object detection from panoramic images. *Sensors*, 19:2622.

## 8 Our GitLab repository

[https://github.com/sidb10/PMI\\_SCNN](https://github.com/sidb10/PMI_SCNN)





## Fiche de Notation PMI

Sujet :

**Spherical Convolution Neural Networks : Application to cars license plate recognition**

Entreprise :

**IPSA**

Tuteur :

**Assia Belbachir et Joana Frontera-Pons**

Etudiants :

<b>Sidney Bakhouché</b>	<b>16,5/ 20</b> Note seuil pour valider : 10/20
<b>Quentin Sarda</b>	<b>16,5/ 20</b> Note seuil pour valider : 10/20
<b>Thomas Schneider</b>	<b>16,5/ 20</b> Note seuil pour valider : 10/20
<b>Sebastien Touche</b>	<b>16,5/ 20</b> Note seuil pour valider : 10/20

Date : 31/01/2020

# FICHE DE NOTATION GROUPE DE RAPPORT DE PMI

*Notation form for Internship Report PMI*

	--	-	+	++
Mise en page (confort de lecture, illustrations...) / <i>Layout (reading confort, illustrations...)</i>	◐	◐	◐	●
Structure du document (équilibre du plan, qualité du sommaire...) / <i>Structure of the document (balance of the plan, quality of summary...)</i>	◐	◐	●	◐
Fiche de synthèse / <i>Page abstract</i>	◐	◐	●	◐
Introduction, Conclusion (bilan technique et apport personnel) / <i>Introduction, Conclusion (technical overview and personal investment)</i>	◐	◐	●	◐
Présentation de l'entreprise (originalité, analyse personnelle, environnement de travail) / <i>Company presentation (originality, personal analysis, working environment)</i>	◐	◐	●	◐
Présentation et analyse du travail réalisé - Contenu technique / <i>Presentation and analysis of the work</i>	◐	◐	◐	●
Orthographe, Grammaire, Style / <i>Spelling, Grammar, Style</i>	◐	◐	◐	●
Résumé, Abstract (+ mots clés, key words) / <i>Summary, Abstract (+key words)</i>	◐	◐	●	◐
Pages complémentaires (Annexes, bibliographie, webographie, sigles et abréviation, index...) / <i>Additional pages (Annex, bibliography, webographie, notes &amp; abbreviations, index...)</i>	◐	◐	●	◐
Connaissances techniques / <i>Technical knowledge</i>	◐	◐	◐	●
Difficulté du sujet / <i>Difficulty of the subject</i>	◐	◐	◐	●

**NOTE FINALE**

**17/ 20**

Note seuil pour valider :  
10/20

Commentaires <i>Comments</i>	Appréciation générale <i>General appreciation</i>
La problématique et l'état de l'art ont été bien positionnées.	


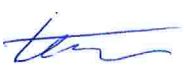


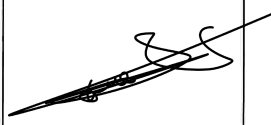


## FICHE DE NOTATION INDIVIDUELLE DE PMI

*Individual Notation form for Internship Report PMI*

Etudiant/Commentaires	Travail /10	Soutenance /10	Moyenne /20
Sidney Bakhouché	8,5	8	16,5
Quentin Sarda	8,5	8	16,5
Thomas Schneider	8,5	8	16,5
Sebastien Touche	8,5	8	16,5

**Signatures :**

Tuteur	Etudiant 1	Etudiant 2	Etudiant 3	Etudiant 4
 Joana Frontera	 Touche Sebastien	 Sidney Bakhouché	 Thomas SCHNEIDER	 Quentin Sarda

# FICHE DE NOTATION GROUPE DE SOUTENANCE DE PMI

*Notation form for Internship Report PMI*

	--	-	+	++
<b>Respect du temps imparti/</b> <i>Respect of time allocated</i>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
<b>Originalité et qualité du support visuel /</b> <i>Originality and quality of audio visual support</i>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>
<b>Facilité d'élocution et clarté de l'expression. Utilisation d'un vocabulaire professionnel /</b> <i>Elocution and clarity of expression. Professional vocabulary</i>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>
<b>Force de conviction et maîtrise du sujet /</b> <i>Force of conviction and knowledge of the subject</i>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>
<b>Qualité de l'exposé en anglais /</b> <i>Quality of the presentation in English</i>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>
<b>Présentation de l'entreprise. Situation du contexte du PMI (QQOQCP) /</b> <i>Company presentation. Place the context of the training</i>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>
<b>Présentation du travail réalisé /</b> <i>Presentation of the work</i>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
<b>Aptitude à l'échange (Pertinence des réponses) et écoute du jury /</b> <i>Question and answers (pertinence of responses) and listening of the jury</i>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>
<b>Niveau de difficulté du PMI (techniques et/ou environnementales) /</b> <i>Difficulty of the internship) Technical and environmental</i>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>

**NOTE FINALE**

**16/ 20**

Note seuil pour valider :  
10/20

<b>Commentaires</b> <i>Comments</i>	<b>Appréciation générale</b> <i>General appreciation</i>