

OpenBCI, ou comment lire dans les pensées en Java

@TrevorReznik

SERLI



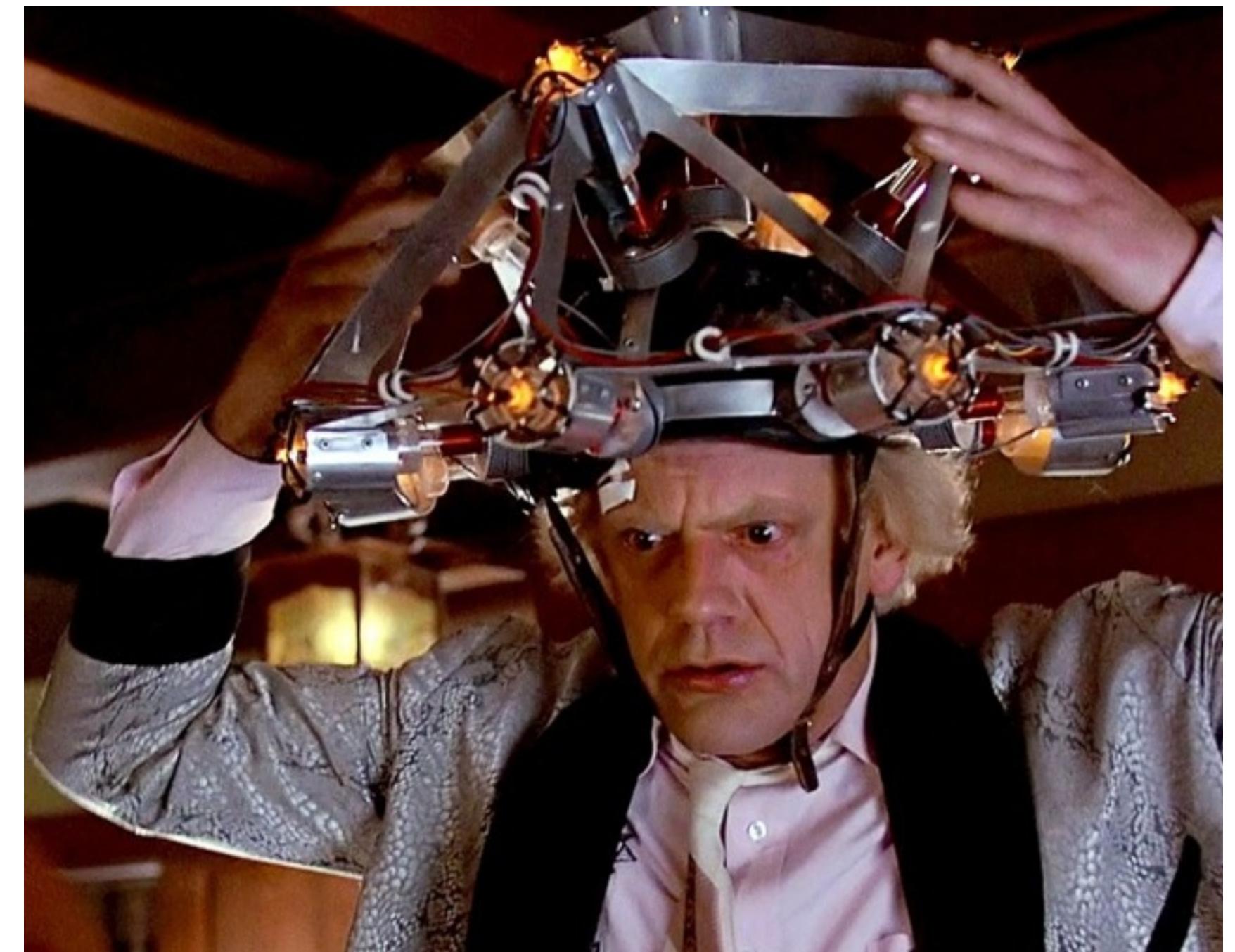
Mathieu ANCELIN

- Développeur @SERLI
- Scala, Java, JS, web & OSS
 - ReactiveCouchbase, Weld-OSGi, etc ...
- Poitou-Charentes JUG
- Membre de l'expert group CDI
- @TrevorReznik



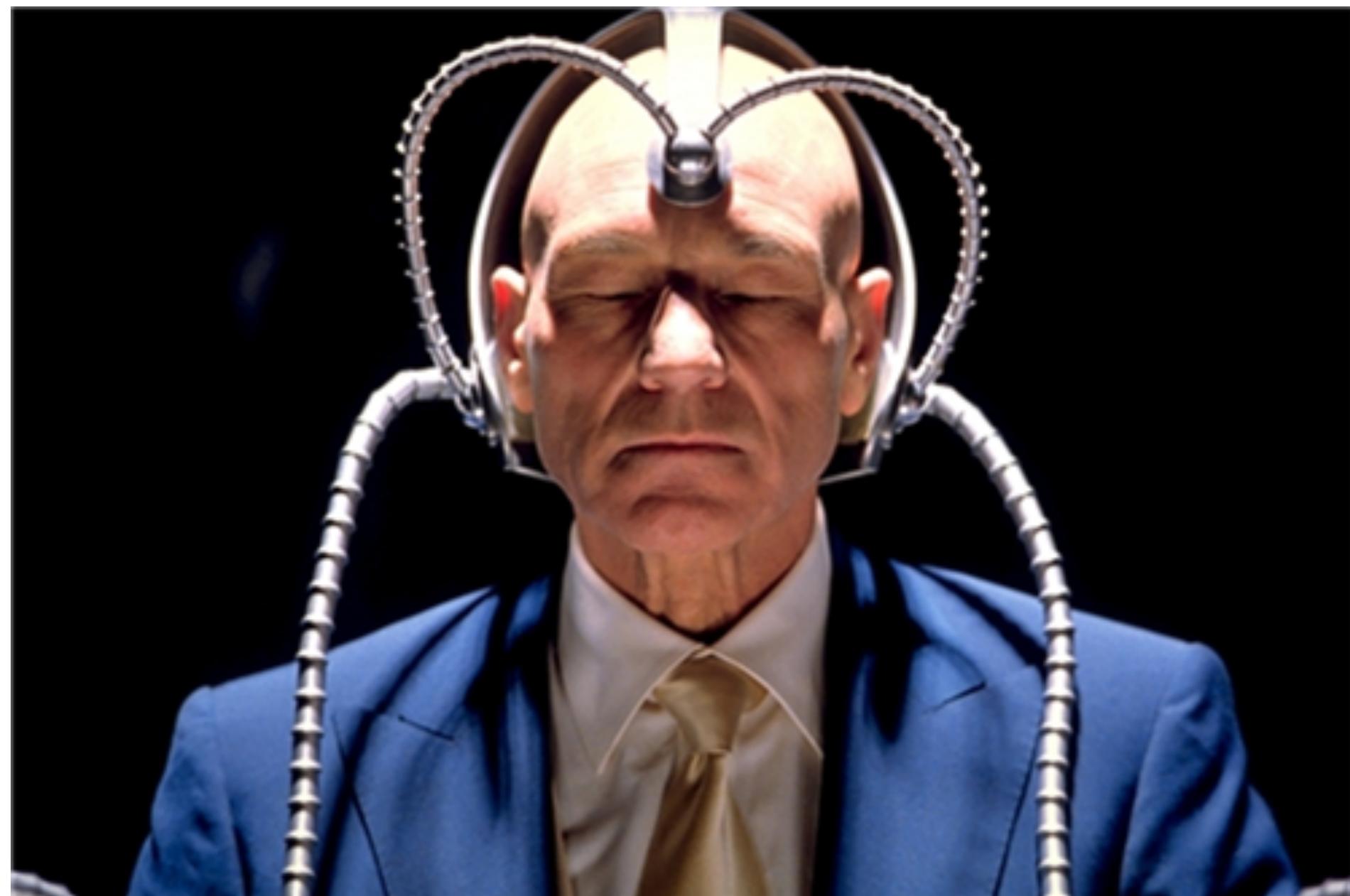
Mathieu ANCELIN

- Développeur @SERLI
- Scala, Java, JS, web & OSS
 - ReactiveCouchbase, Weld-OSGi, etc ...
- Poitou-Charentes JUG
- Membre de l'expert group MVC
- @TrevorReznik



Mathieu ANCELIN

- Développeur @SERLI
- Scala, Java, JS, web & OSS
 - ReactiveCouchbase, Weld-OSGi, etc ...
- Poitou-Charentes JUG
- Membre de l'expert group MVC
- @TrevorReznik



Serli

- Société de conseil et d'ingénierie du SI
- 70 personnes
- Java, cloud, mobilité
- Contribution à des projets OSS
- 10% de la force de travail sur l'OSS
- Membre du JCP
- www.serli.com @SerliFr



Disclaimer



Disclaimer

OpenBCI ne permet PAS de lire
dans les pensées !!!!



Disclaimer

Je n'ai pas fait de bio depuis plus de 12
ans



DEMO WILL FAIL

I GUARANTEE IT

Background

- L'idée de créer un lien direct entre le cerveau humain et un appareil électronique est apparue au milieu des années 60
- Lancement d'un programme par le DoD américain pour aider les pilotes de chasse à piloter leur appareil.
- Devait aider à faire baisser la charge de travail mental pour le pilote
- Technologie pas assez sophistiquée, projet annulé
- Programme annulé au bout de quelques années
- point de base de toutes les recherches sur le sujet

BCI

- Brain Computer Interface
- Système permettant à une personne de contrôler le monde extérieur sans aucune activité musculaire
- Les entrées du système sont les impulsions électro-chimiques directement enregistrées depuis le cerveau
 - reconnaissance de patterns dans ces impulsions
 - association de commandes à ces patterns
 - De manière générale, basé sur 3 étapes

Enregistrement du signal

- Ces appareils sont souvent basés sur des électro-encéphalogrammes (EEG)
 - c'est la méthode la plus simple
 - c'est la méthode la moins chère
 - c'est la méthode la moins invasive
- Capte les impulsions
- les amplifie, les filtre
- les numérise



Pré-processing

- Phase permettant de transformer le signal pour qu'il soit plus simple à traiter
 - filtrage
 - réduction du bruit
 - composition d'inputs
 - etc ...

Classification

- Phase permettant de transformer le signal en catégories
- A partir de ces catégories, le système va pouvoir extraire des commandes
- Souvent, les systèmes de classification sont basé sur de l'auto apprentissage
 - on entraîne le système pour qu'il puisse reconnaître les patterns représentant les commandes
 - chaque personne est unique, il est difficile de généraliser des patterns précis associés à des commandes
 - par exemple, des réseaux de neurones

Le projet OpenBCI

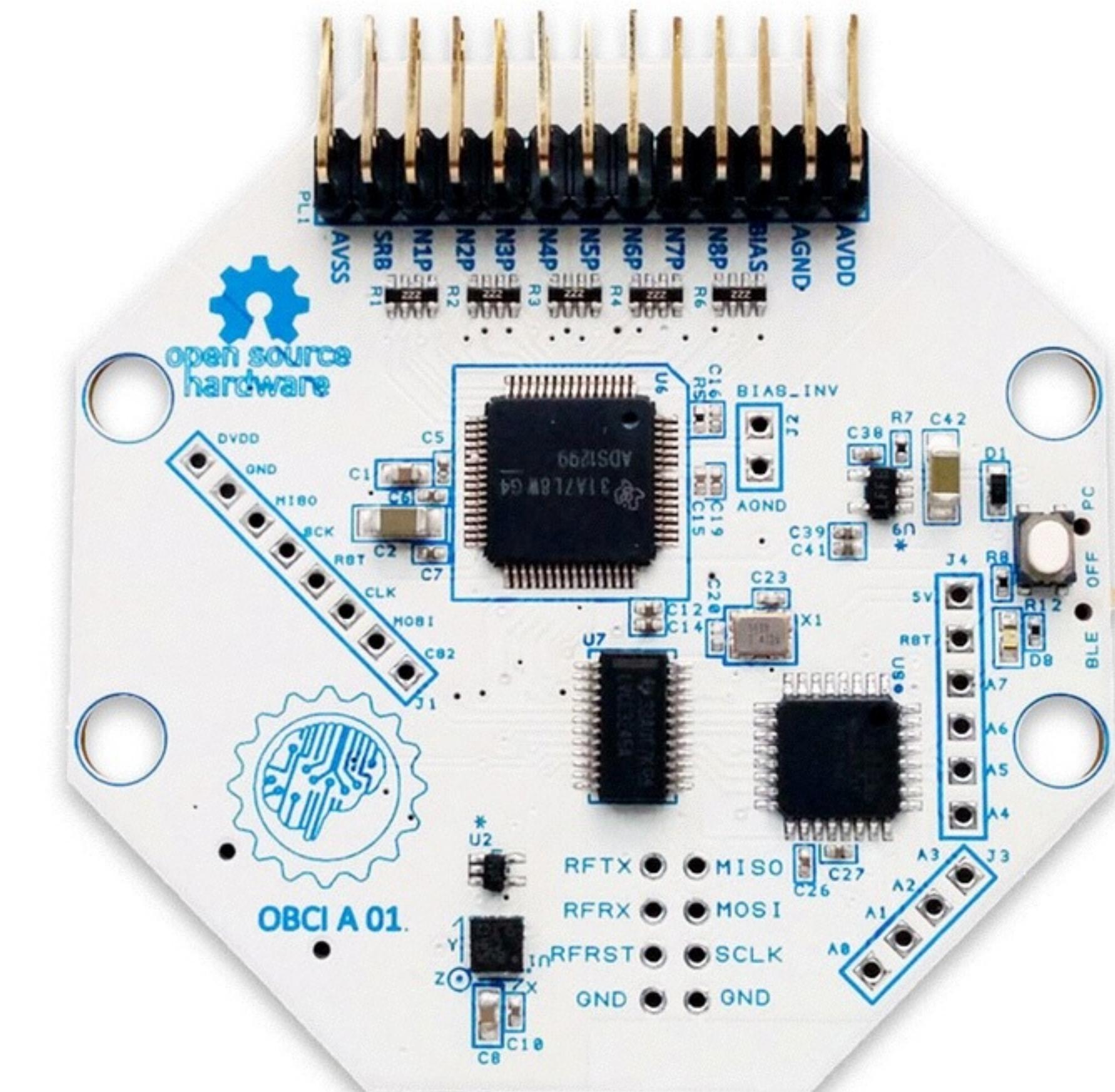


Le projet OpenBCI

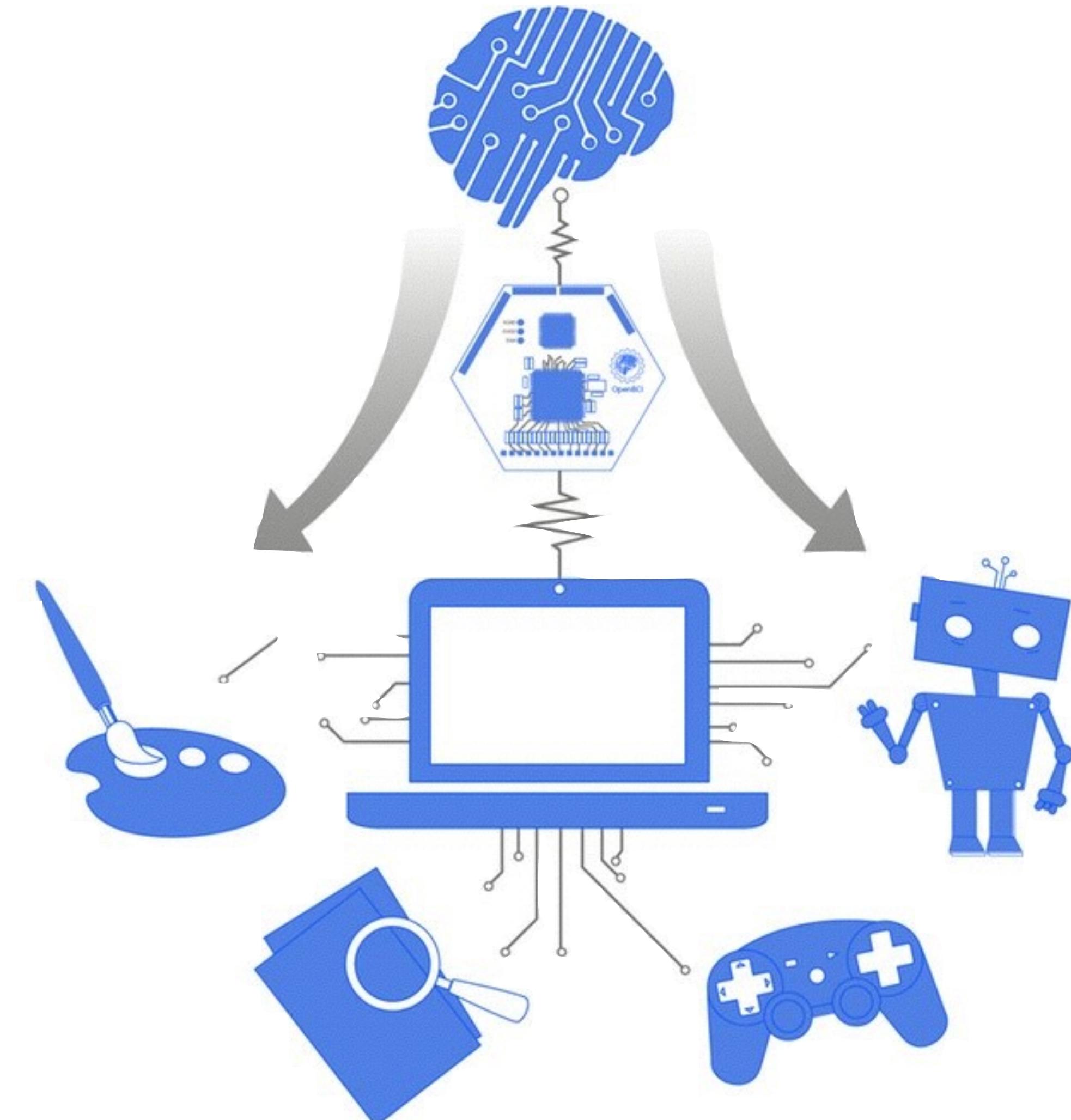
- Projet lancé sur Kickstarter en décembre 2013
 - Joel Murphy
 - Conor Russomanno
 - backé par 947 personnes pour la somme 215 438 \$

« A customizable and fully open brain-computer interface platform that gives you access to high-quality brain wave data. » - Joel Murphy

Le projet OpenBCI



Le projet OpenBCI



Le projet OpenBCI

Unveiled at Maker Faire NYC
Arduino Shield



V1
SEPT 2013

Factory Prototype Improved Design
Arduino Shield



V2
NOV 2013

Programmable over Bluetooth LE
ATmega 328 Integrated
Arduino Compatible
Local SD card



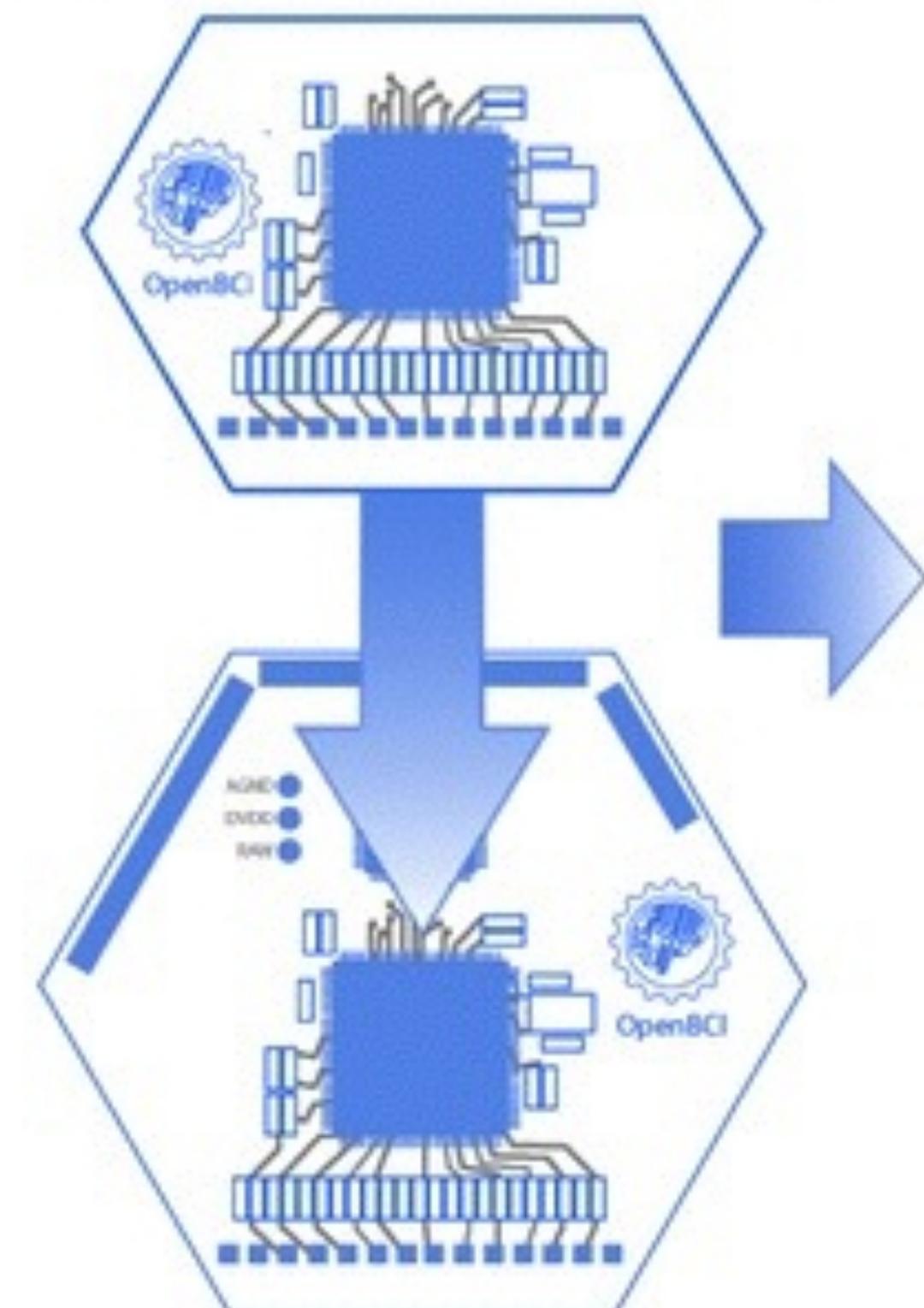
V3
APRIL 2014

OPENBCI EVOLUTION

Le projet OpenBCI

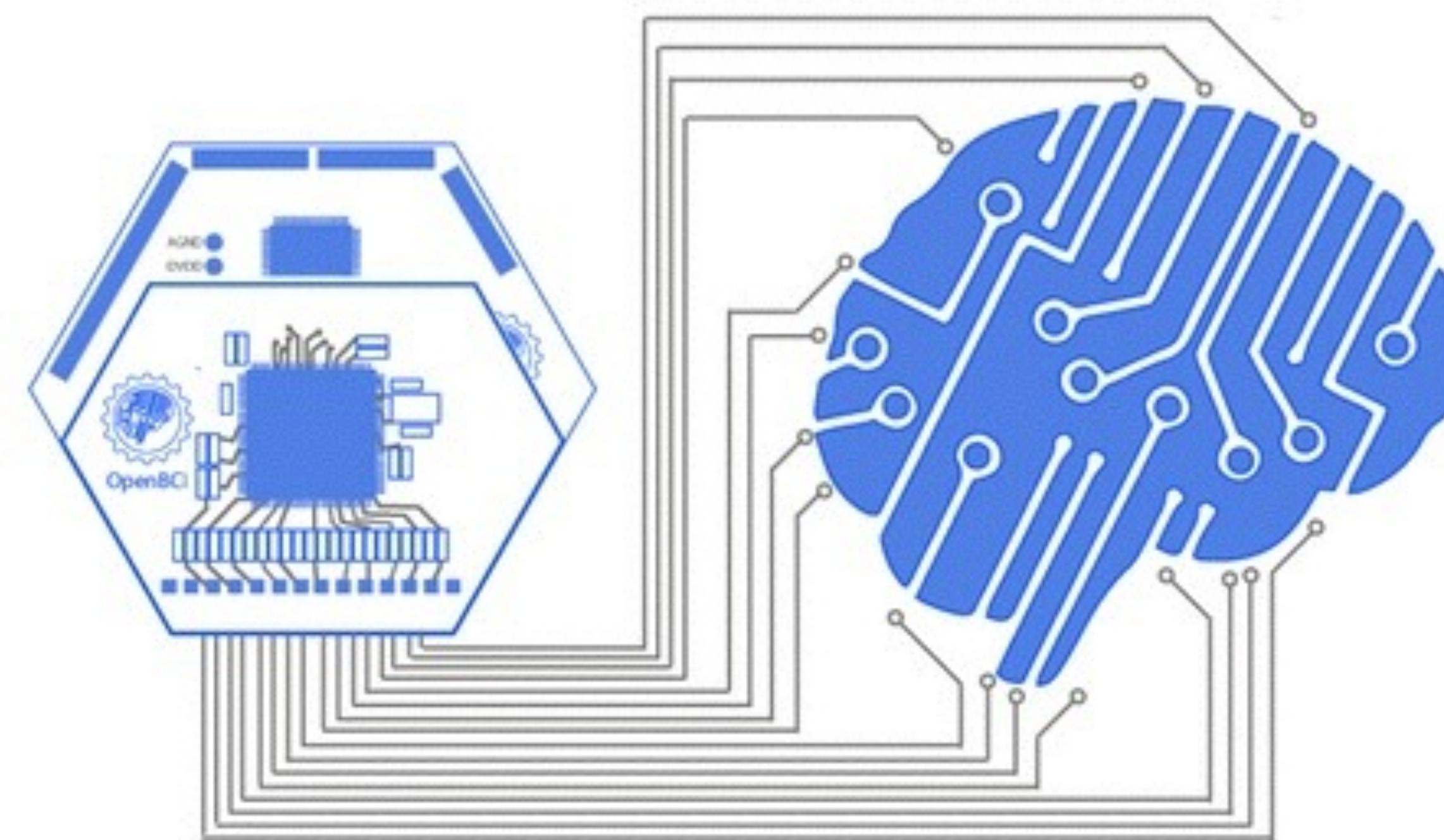
DEVOXX France

Daisy-Chain Module
8 Electrode Inputs



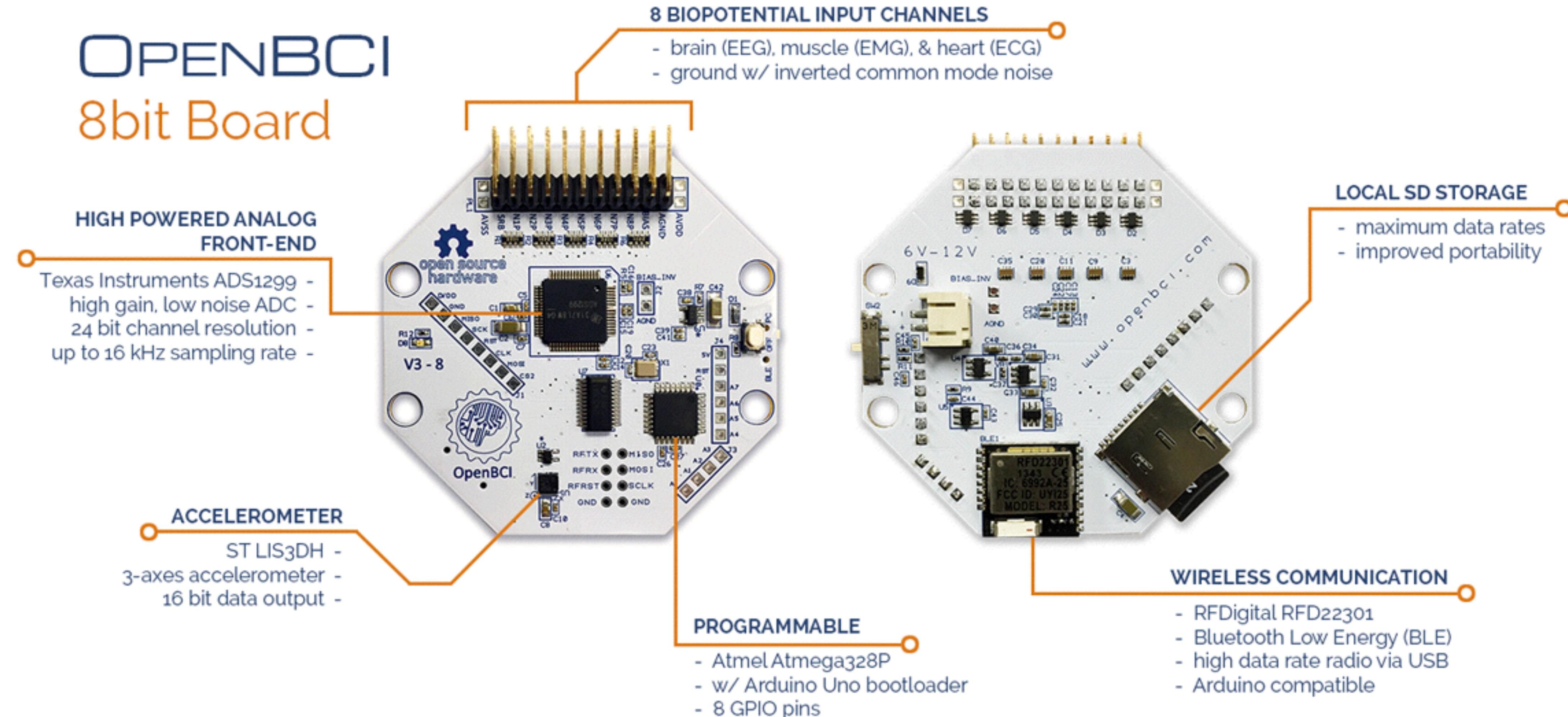
OpenBCI Board
8 Electrode Inputs

Daisy-Chain Setup
16 Electrode Inputs

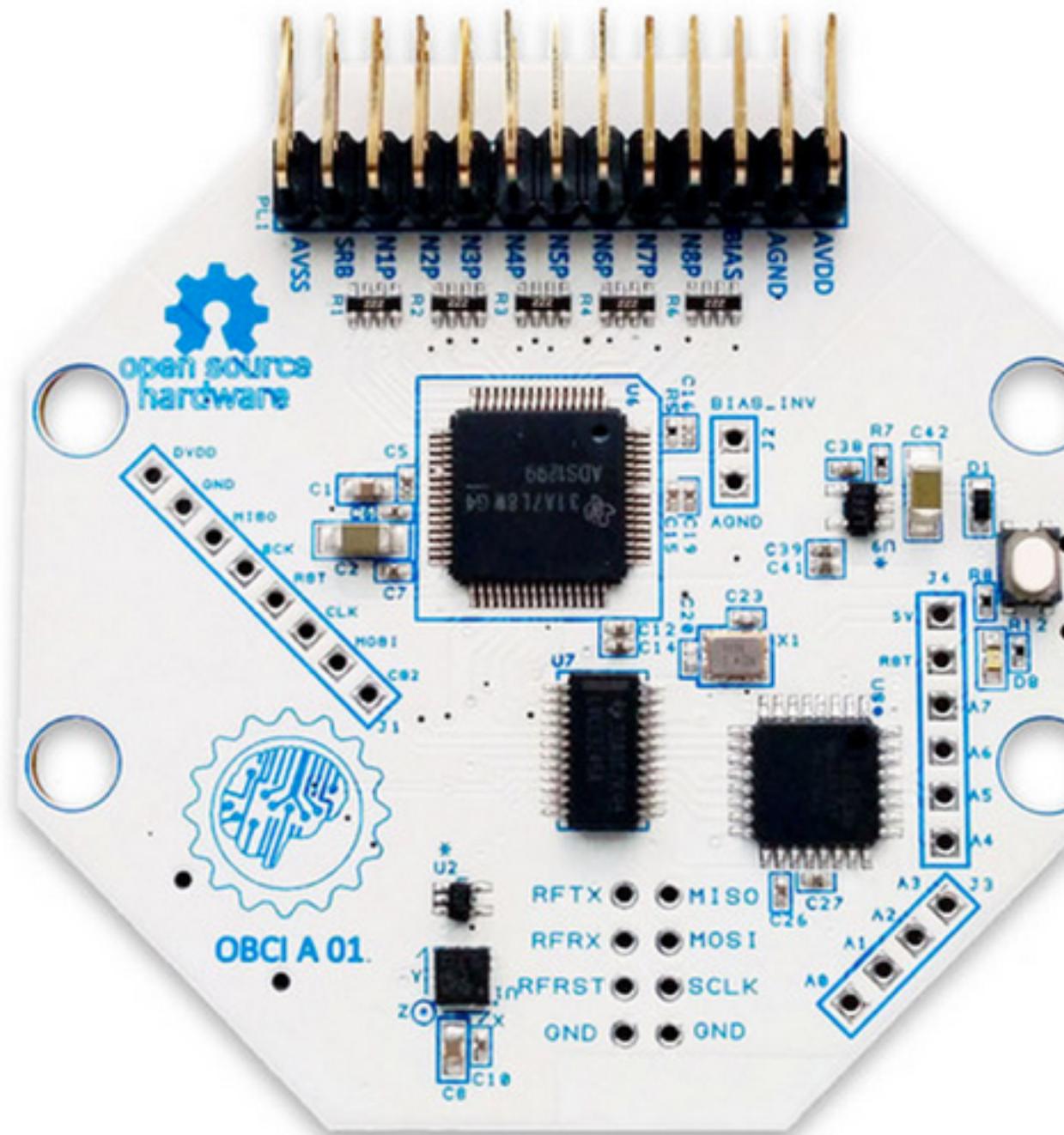


Le projet OpenBCI

OPENBCI 8bit Board



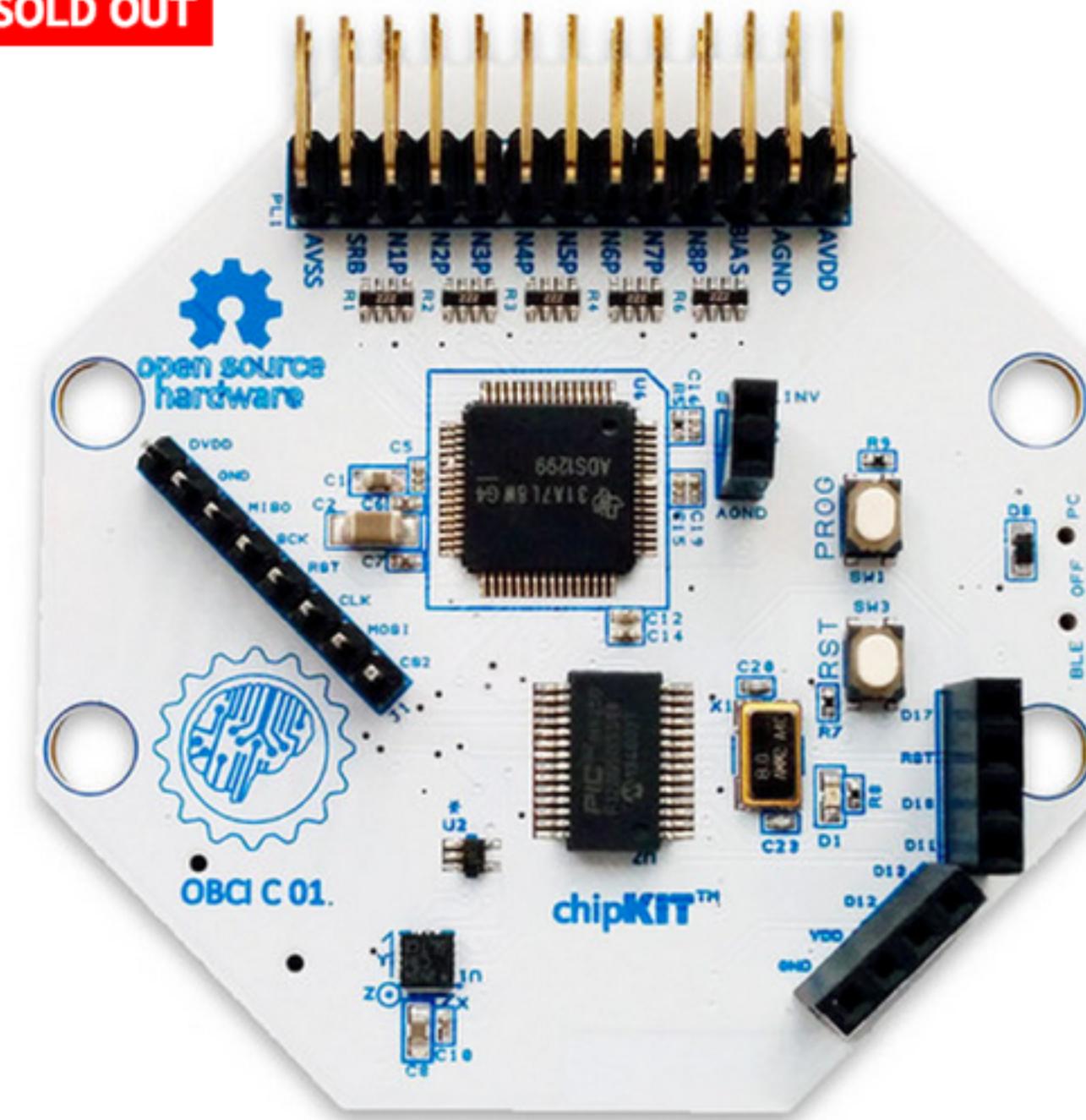
Le projet OpenBCI



OpenBCI 8-bit Board Kit (Arduino™-compatible)

\$ 449.99

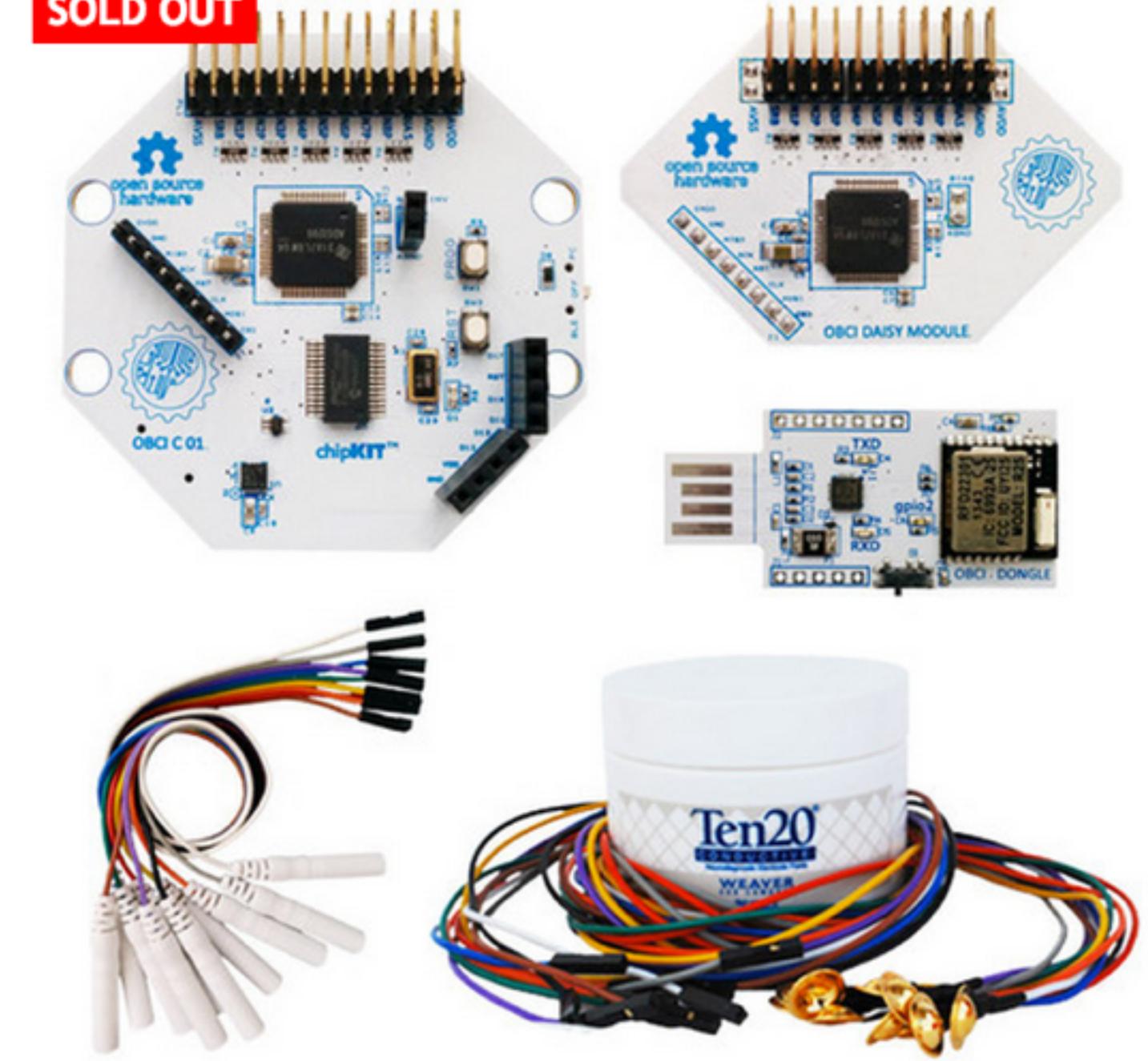
SOLD OUT



OpenBCI 32-bit Board Kit (chipKIT™-compatible)

\$ 449.99

SOLD OUT



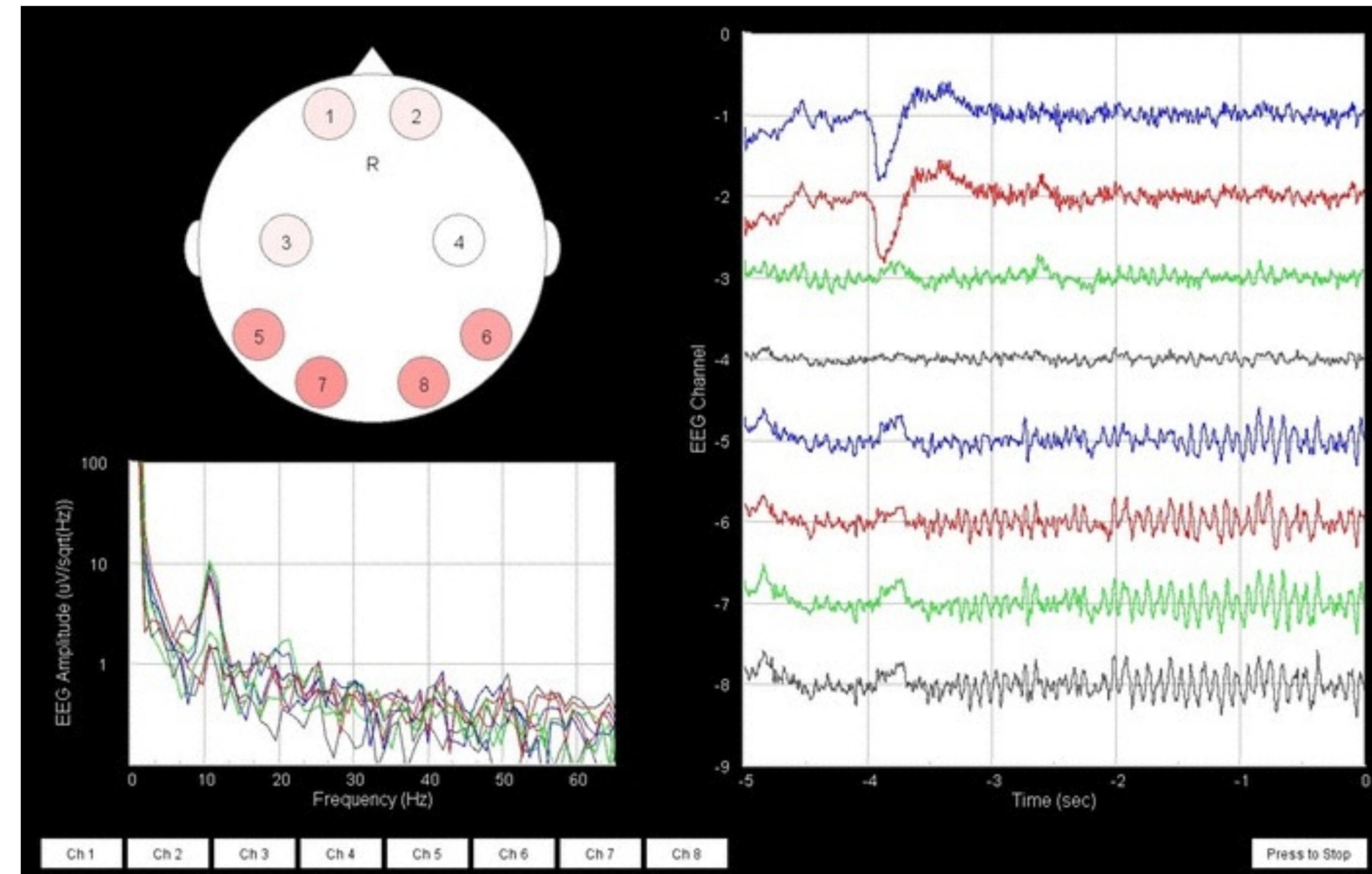
OpenBCI 16-channel R&D Kit

\$ 799.99

Le projet OpenBCI



Les outils



Demo

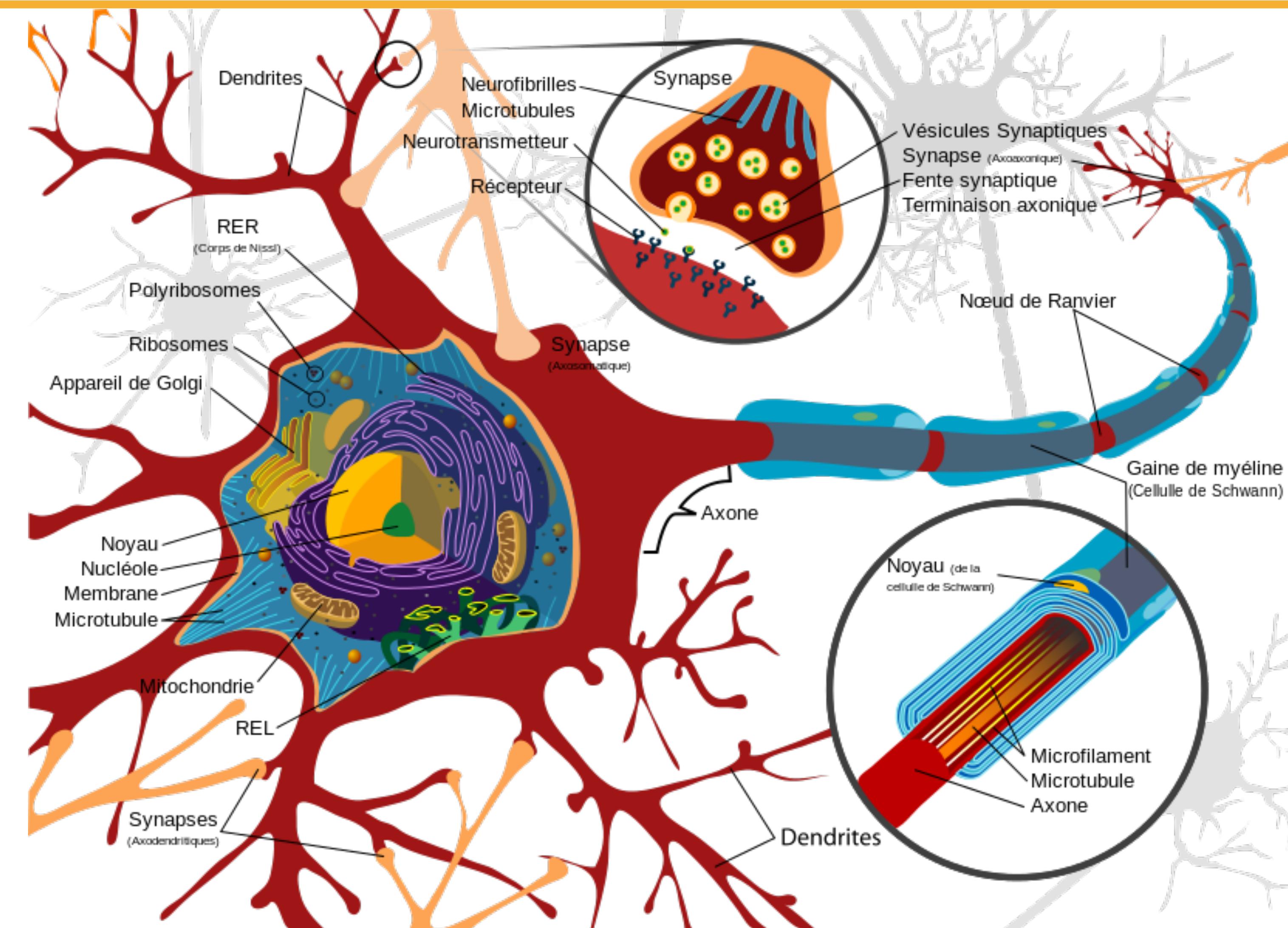


Un peu de théorie

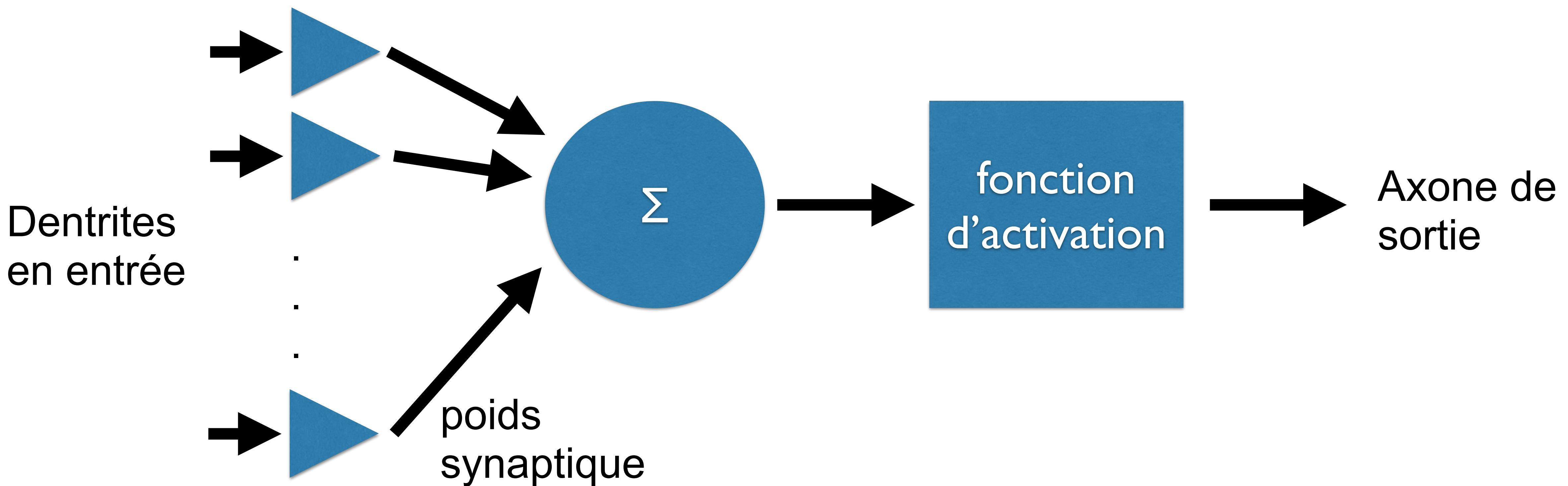


Modèle d'un neurone

DEVOVOX France



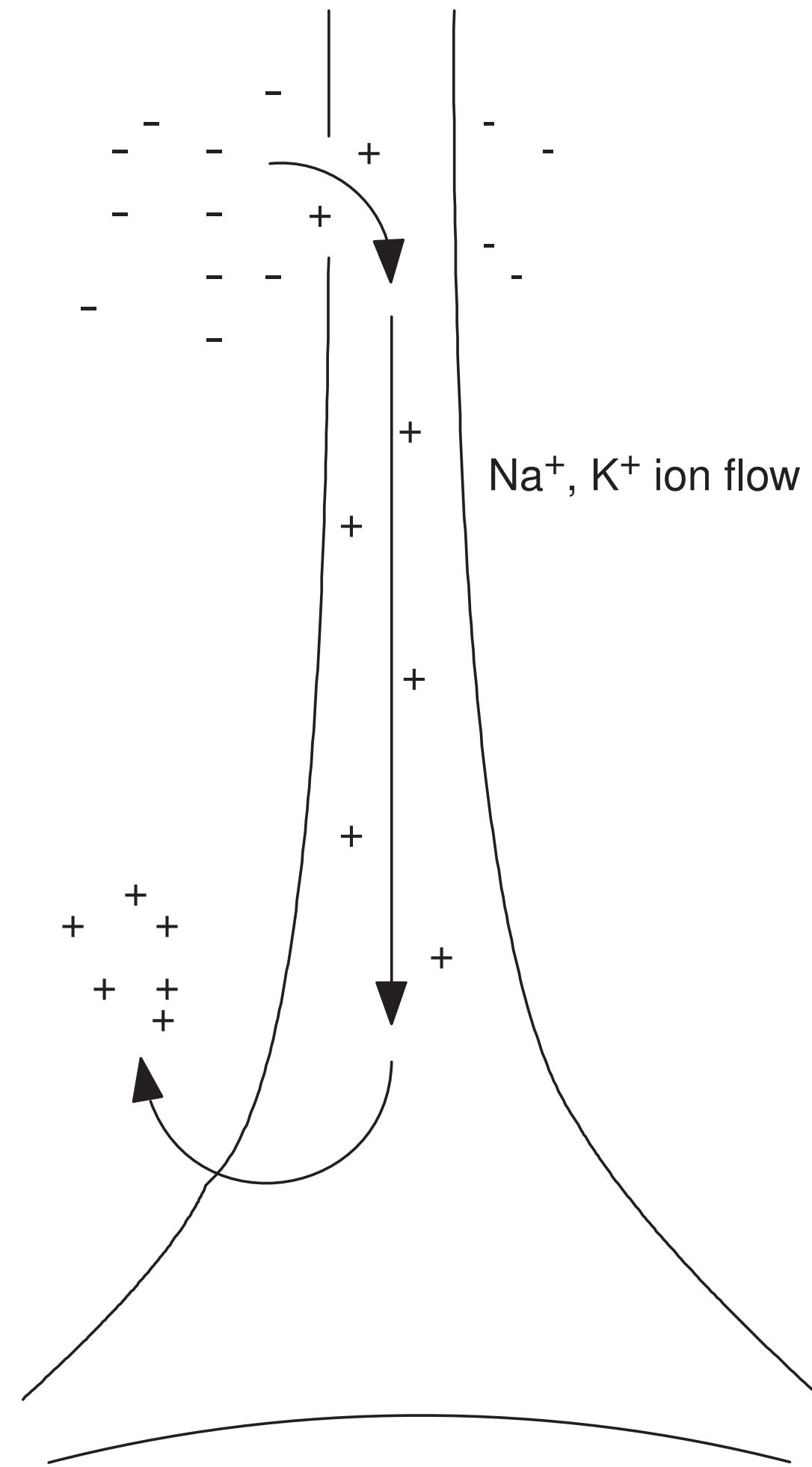
Modèle d'un neurone



Modèle d'un neurone

- Unité de processing autonome
- spécialisée dans la réception et l'émission de signaux électro-chimiques
- accepte en entrée les résultats d'autres cellules à travers les dendrites et les somme
- si l'addition arrive jusqu'à un certains seuil propre au neurone
 - le neurone « fait feu » et envoie un signal de sortie à travers son axone

Modèle d'un neurone



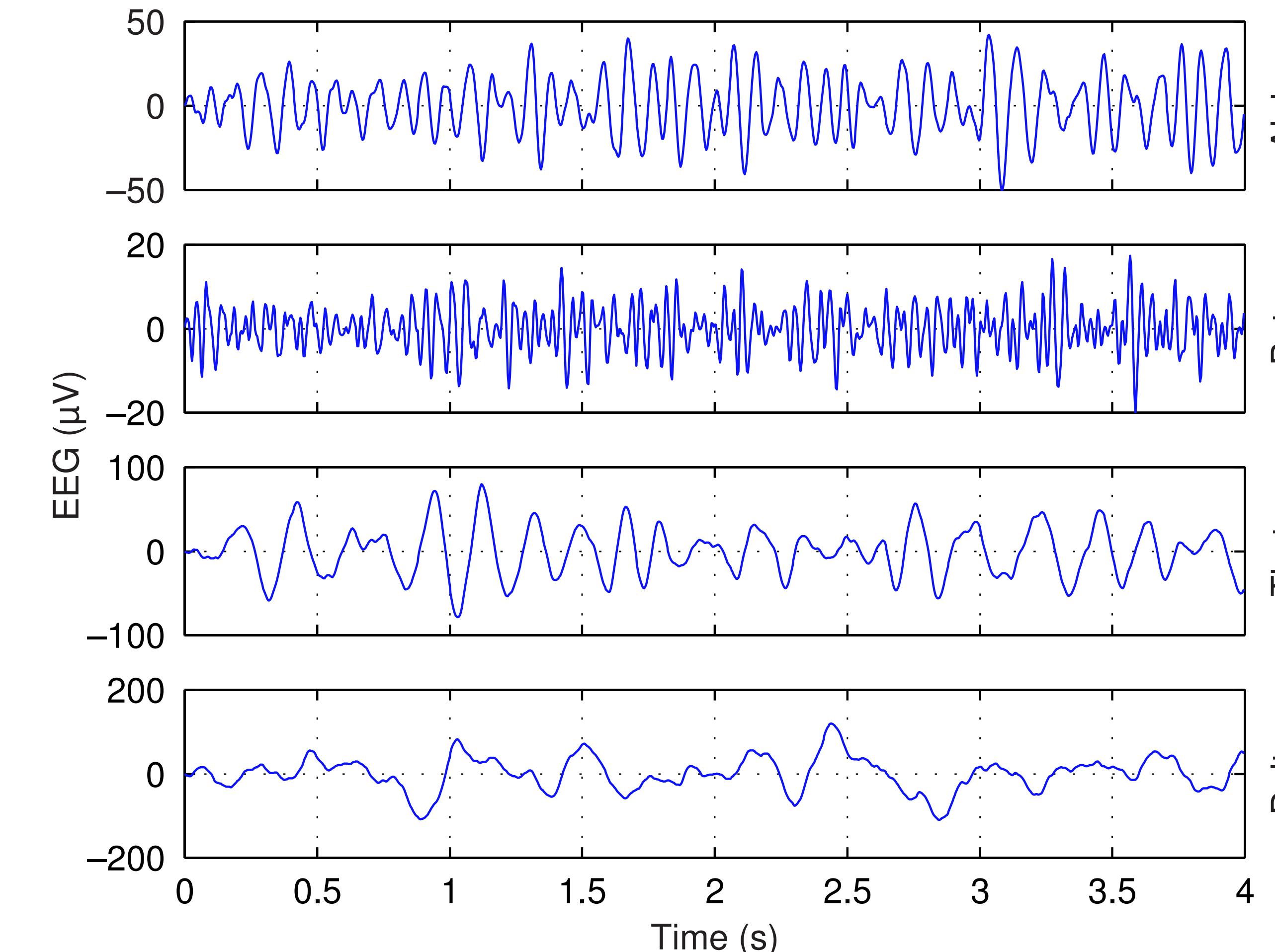
Modèle d'un neurone

- Composés majoritairement de membranes semi perméables et de fluides conducteurs
 - fluide composé majoritairement de potassium et de sodium
 - contrairement à l'extérieur de la cellule
- En cas de stimulus
 - la perméabilité de la membrane change
 - les ions passent à travers
 - chargement de la cellule
 - une protéine fait sortir du potassium de la cellule peu de temps après pour faire baisser le potentiel

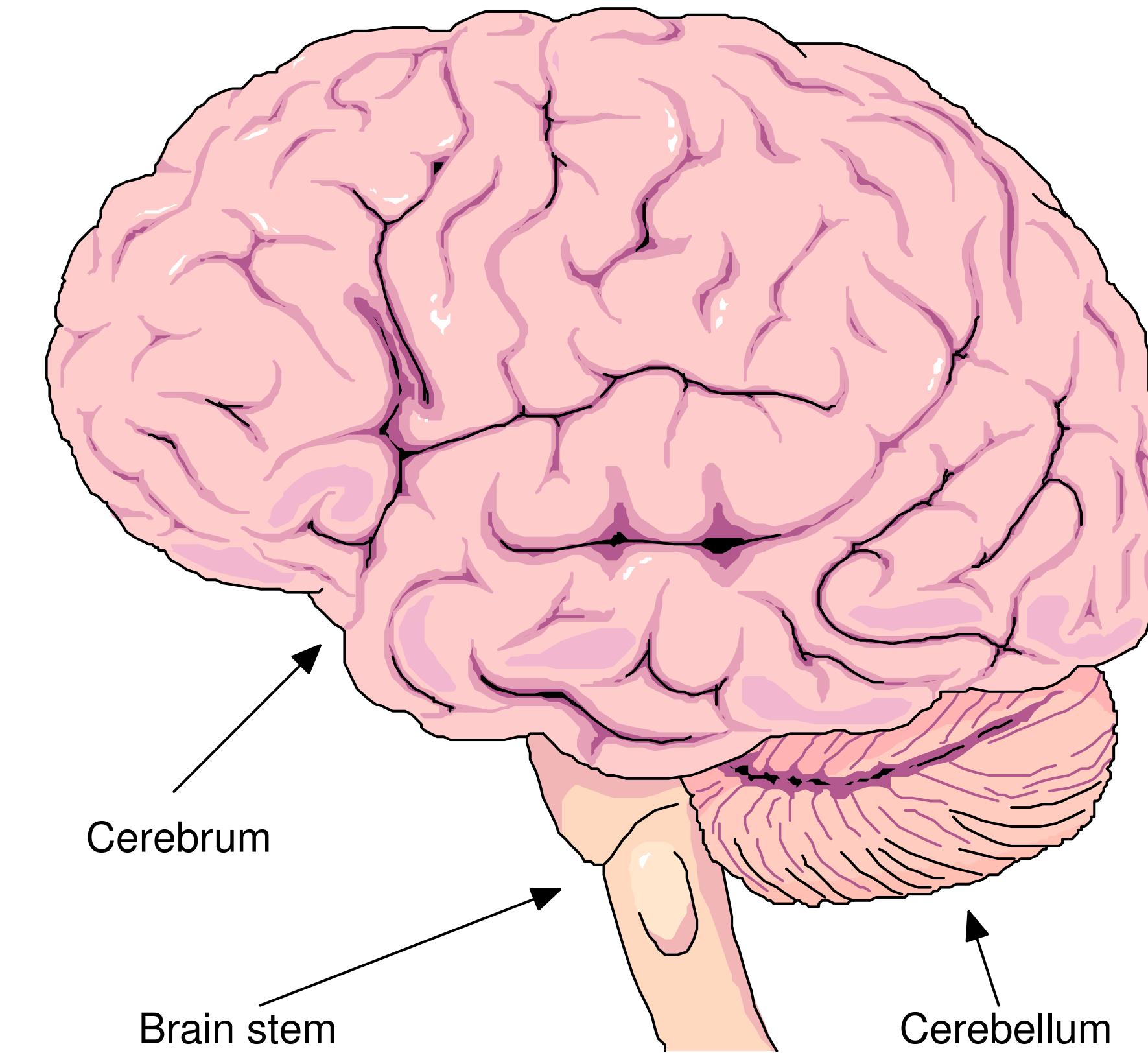
Les rythmes du cerveau

- Ondes Alpha
 - entre 8 et 13 Hz, lorsque le sujet se repose mais est éveillé, les yeux fermés
- Ondes Beta
 - entre 13 et 30 Hz, lorsque le sujet ouvre brusquement les yeux ou entre dans une activité mentale
- Ondes Theta
 - Entre 4 et 8 Hz, stress émotionnel et frustration
- Ondes Delta
 - Entre 0.5 et 4 Hz, sommeil profond

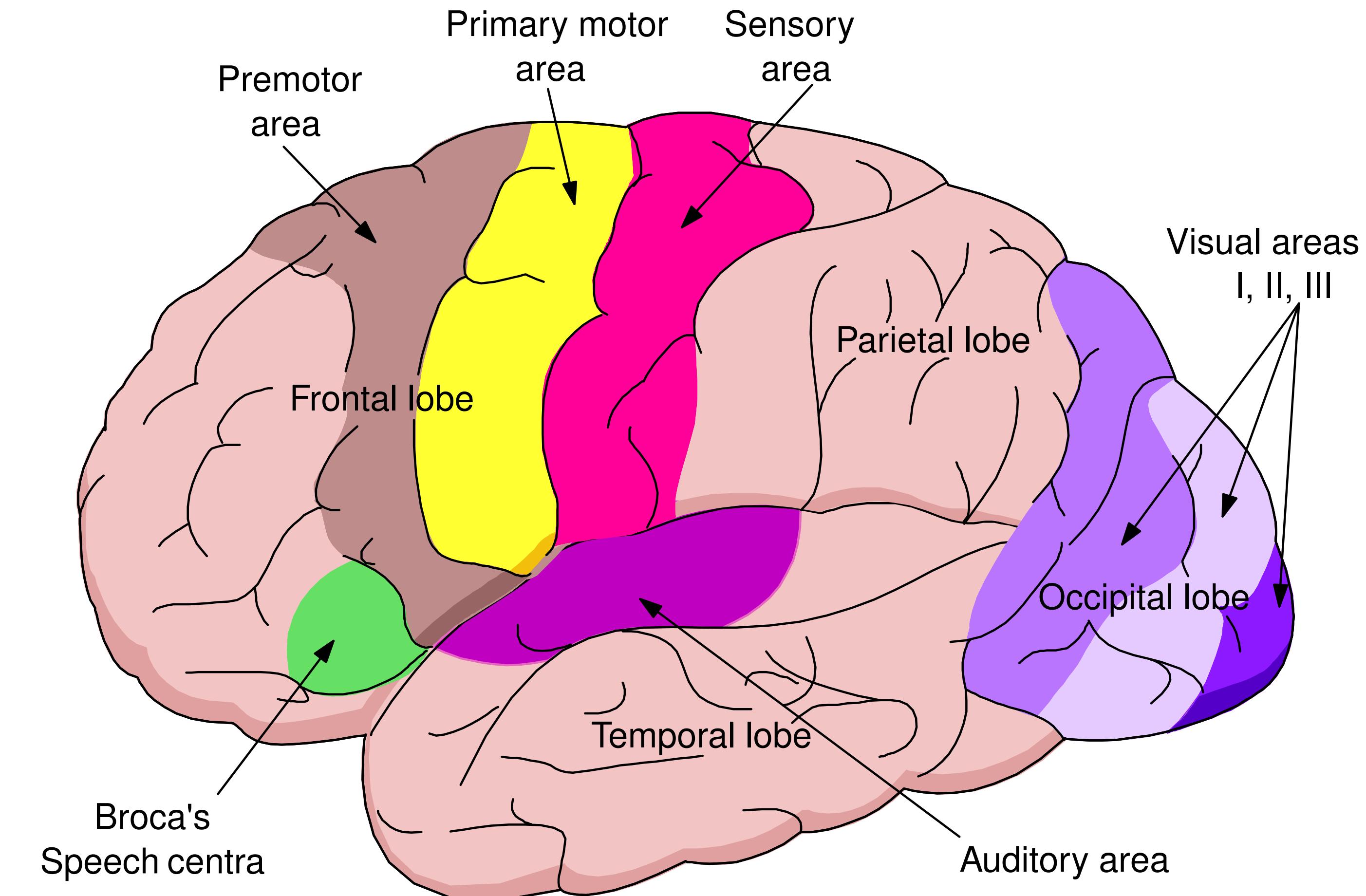
Ondes



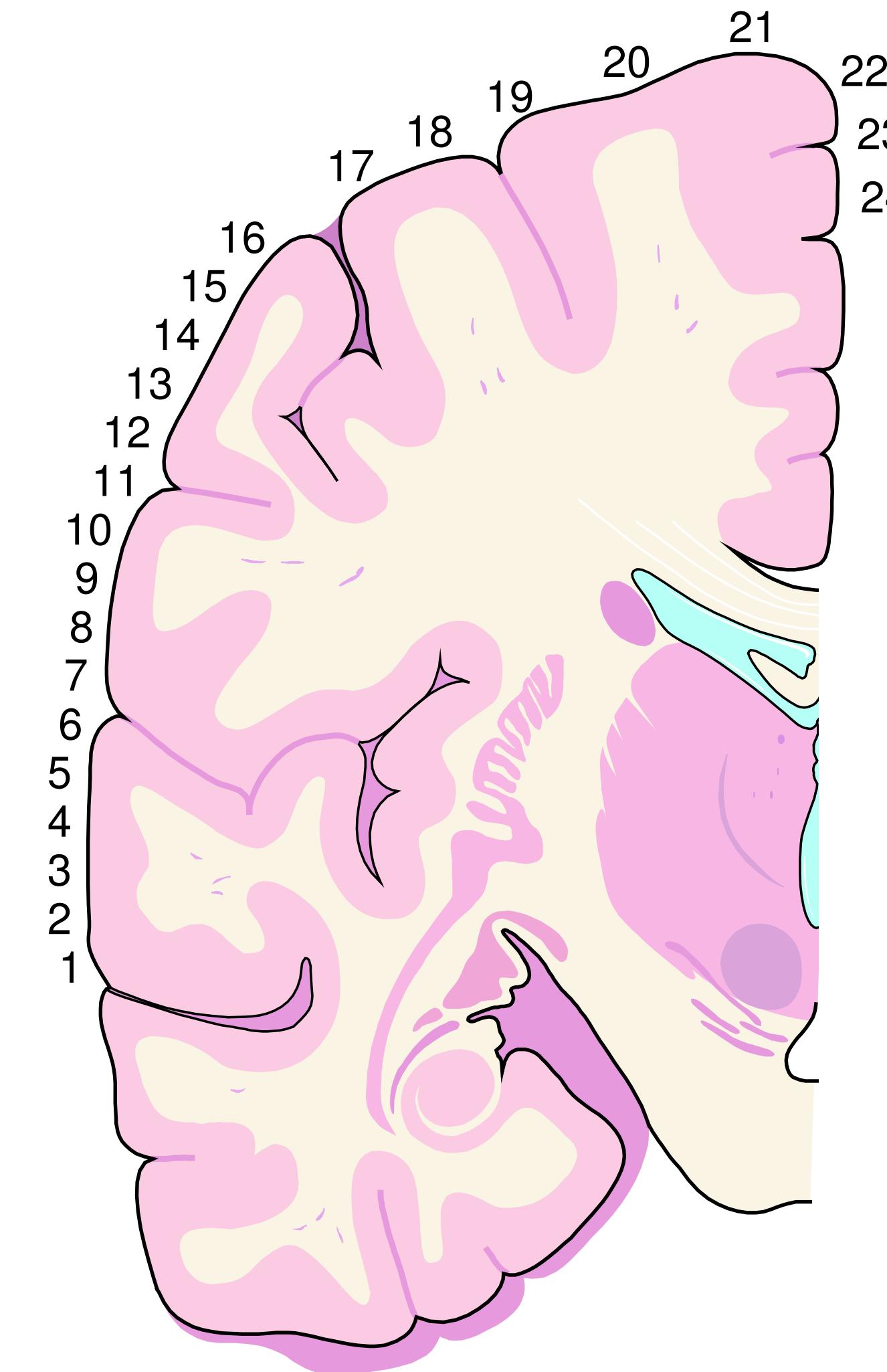
Le cerveau humain



Le cerveau humain

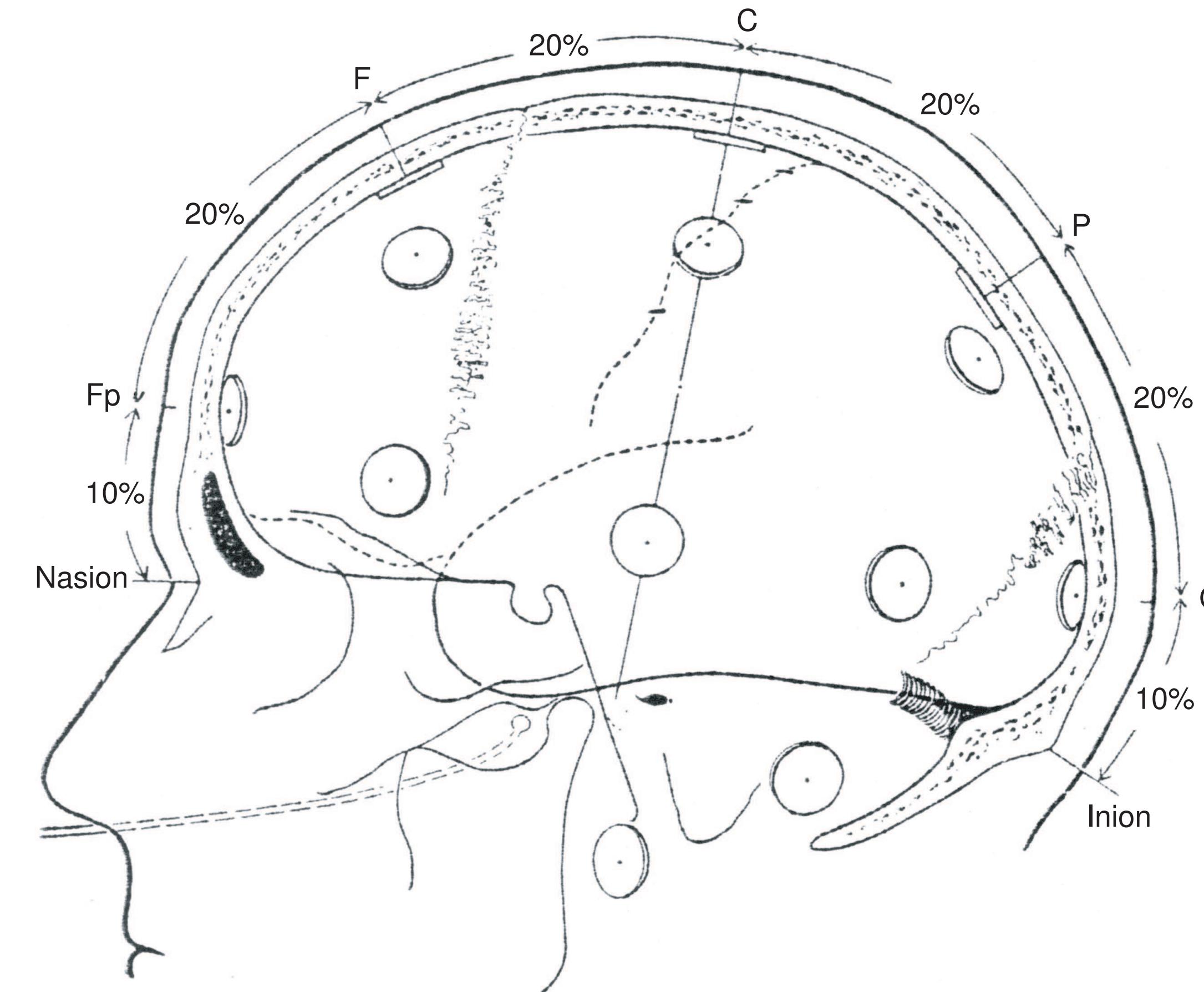


Le cerveau humain

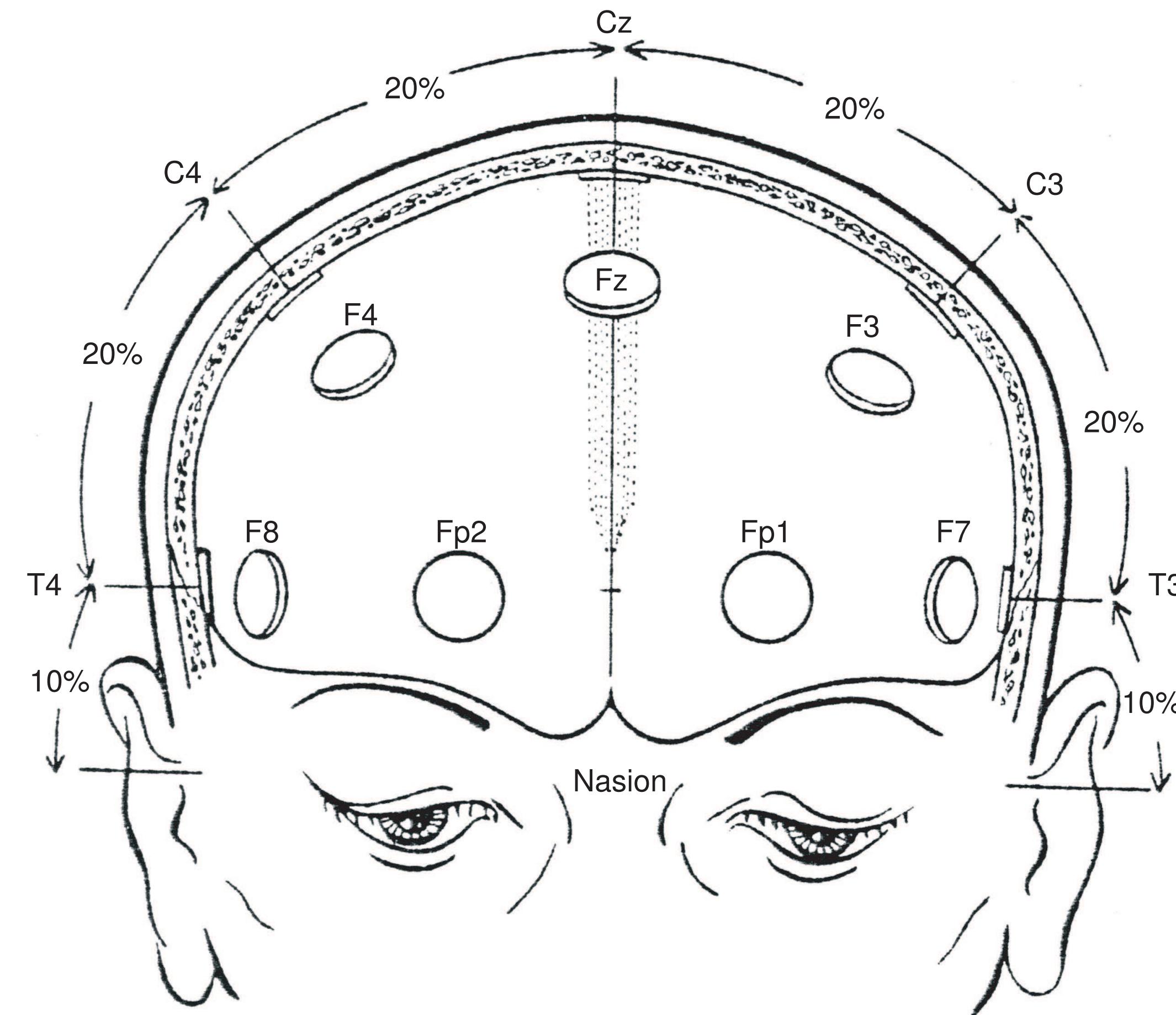


- 1 – chewing
- 2 – swallowing
- 3 – tongue
- 4 – teeth, jaw
- 5 – speech
- 6 – lips
- 7 – face
- 8 – eyes
- 9 – forehead
- 10 – neck
- 11 – thumb
- 12 – index finger
- 13 – middle finger
- 14 – ring finger
- 15 – little finger
- 16 – hand
- 17 – wrist
- 18 – elbow
- 19 – shoulder
- 20 – trunk
- 21 – hip
- 22 – knee
- 23 – ankle
- 24 – toes

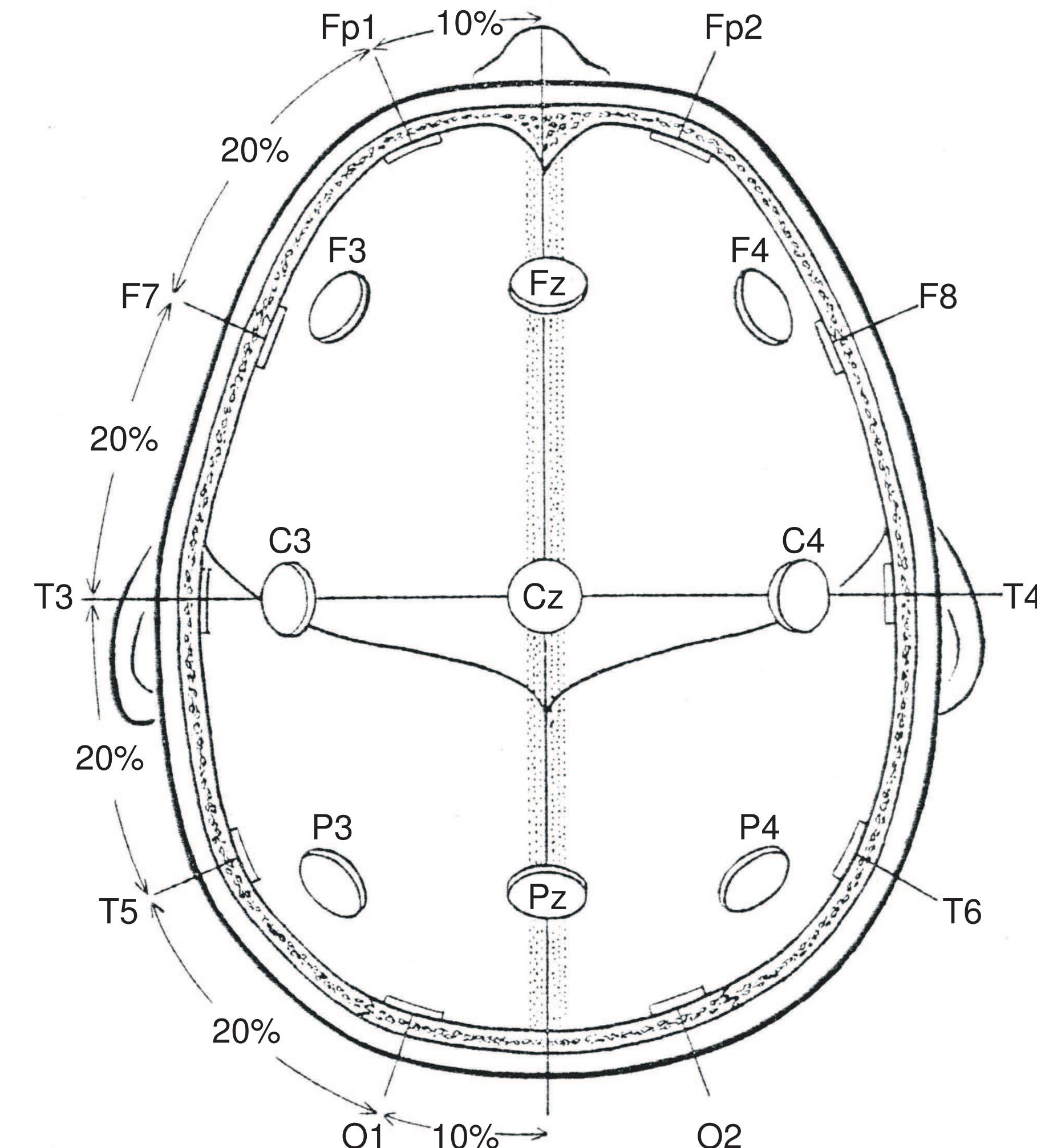
Position des électrodes



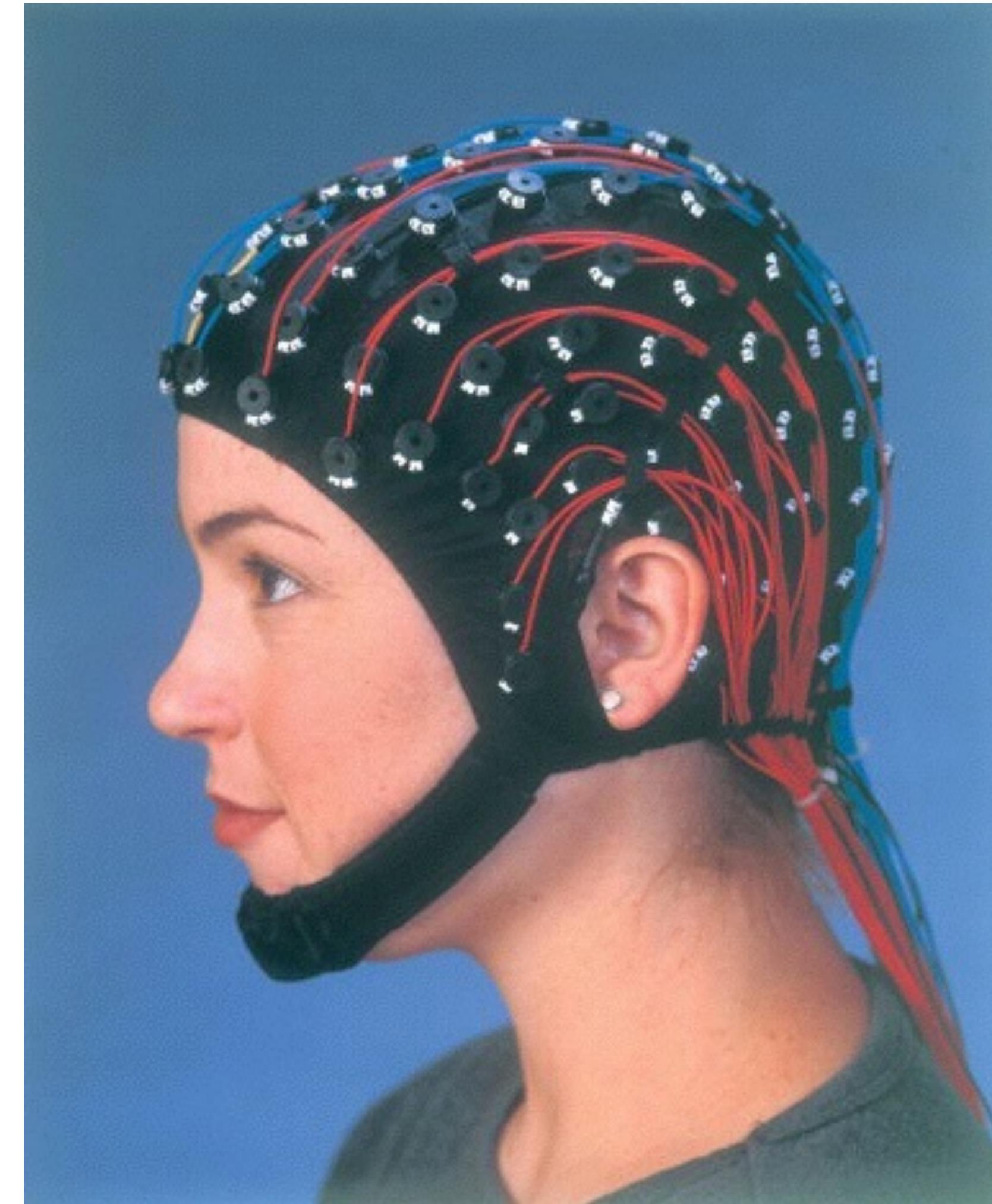
Position des électrodes



Position des électrodes



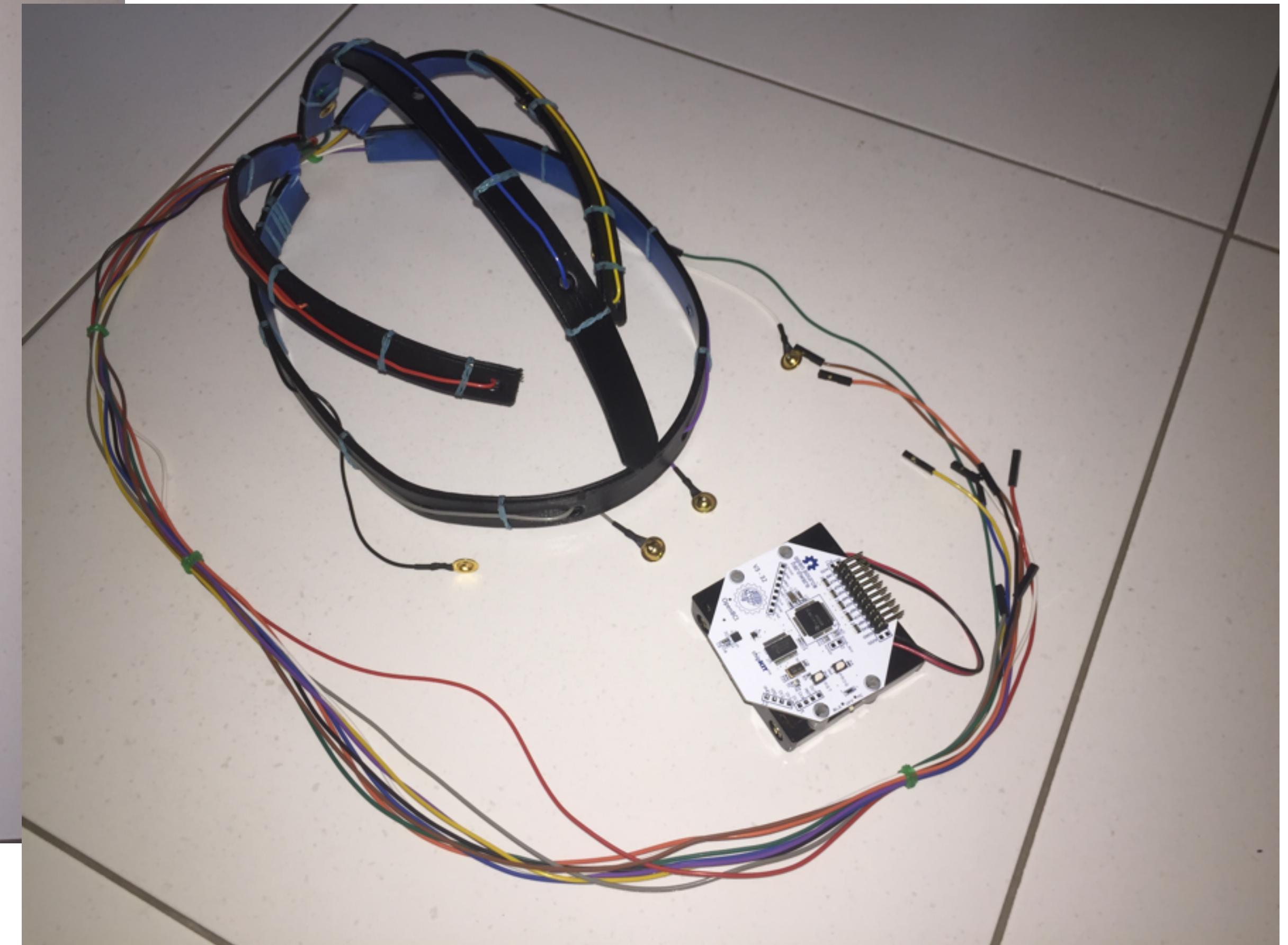
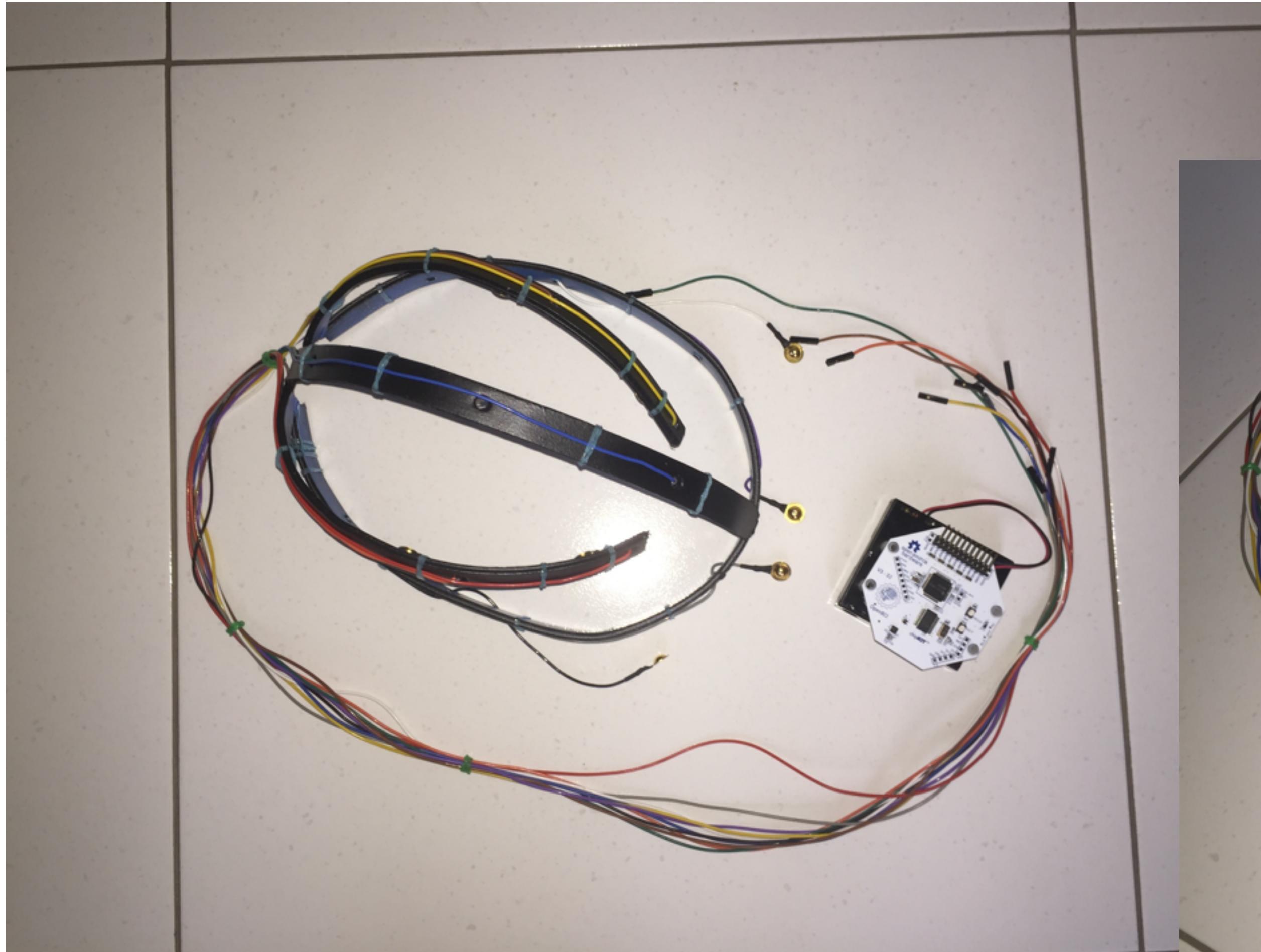
Position des électrodes



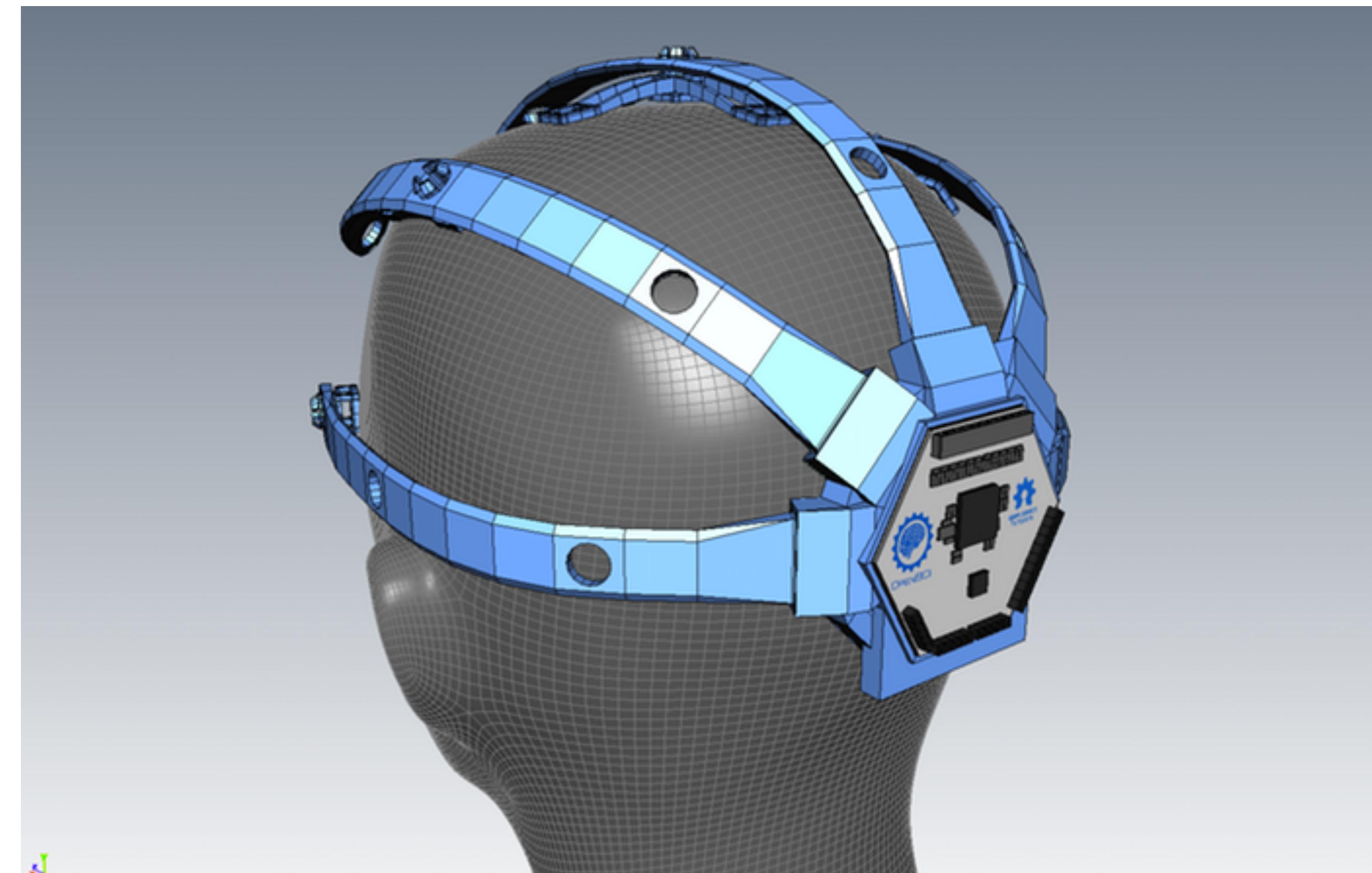
L'environnement de démo



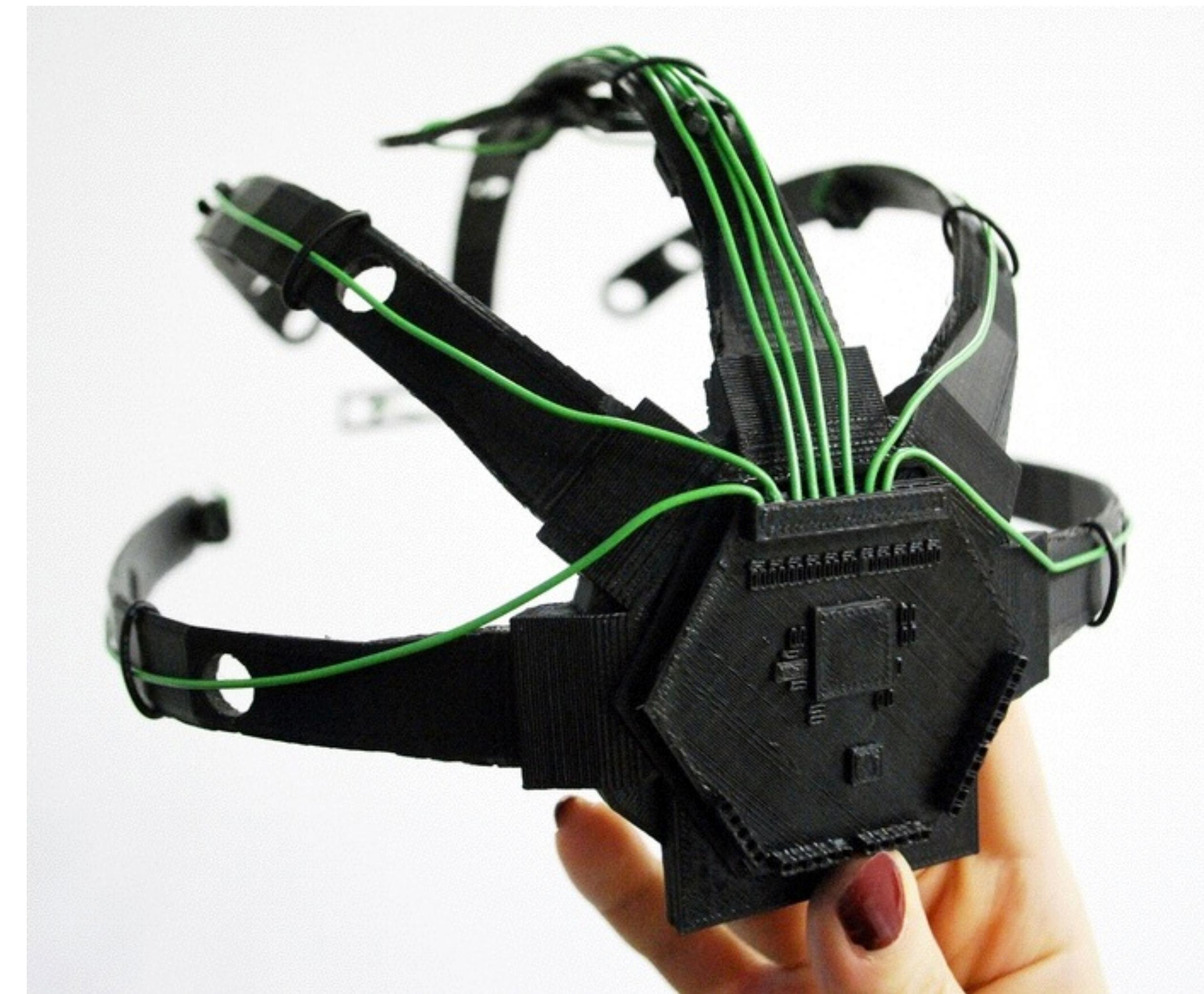
Le cérébro



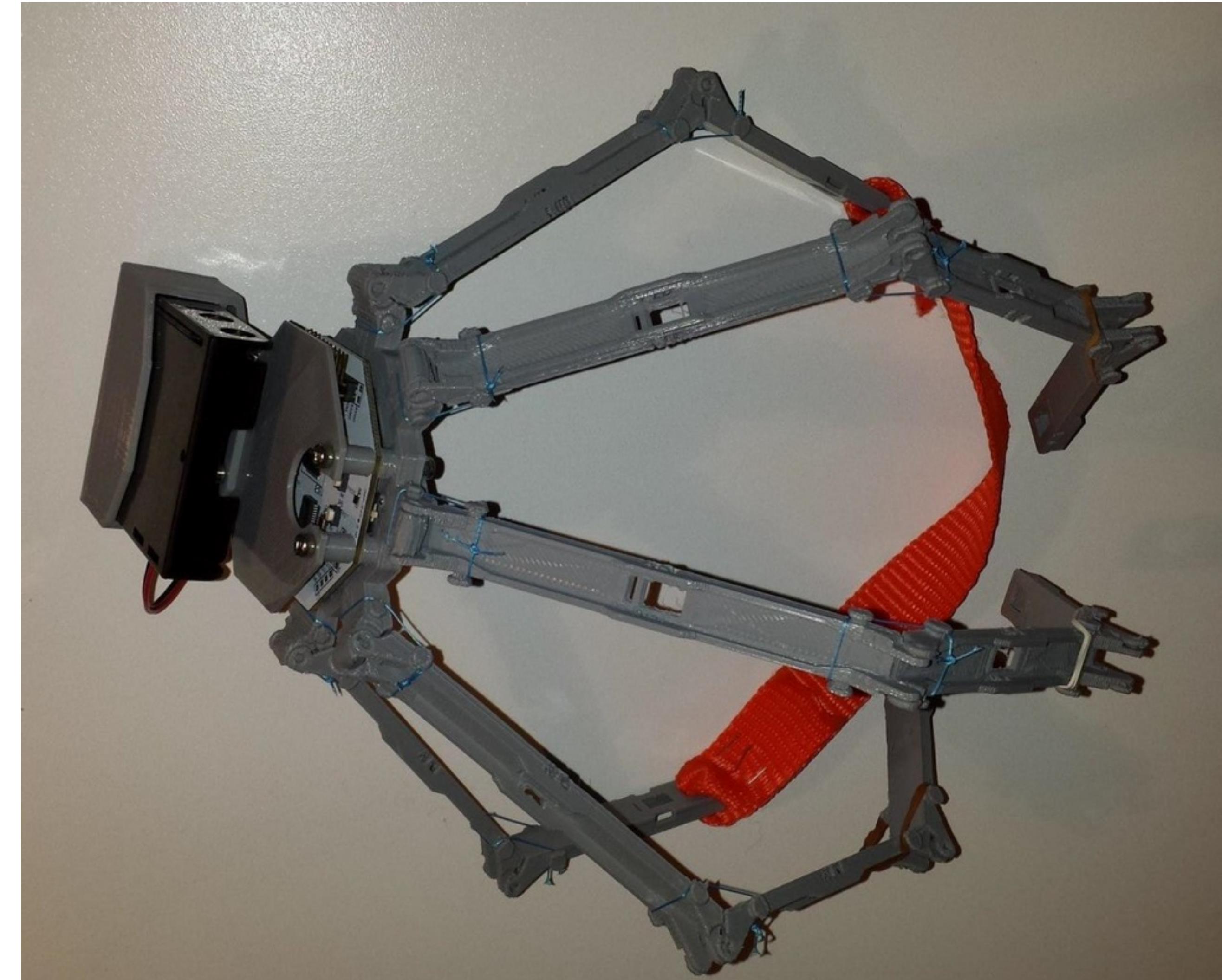
Imprimer son propre casque



Imprimer son propre casque



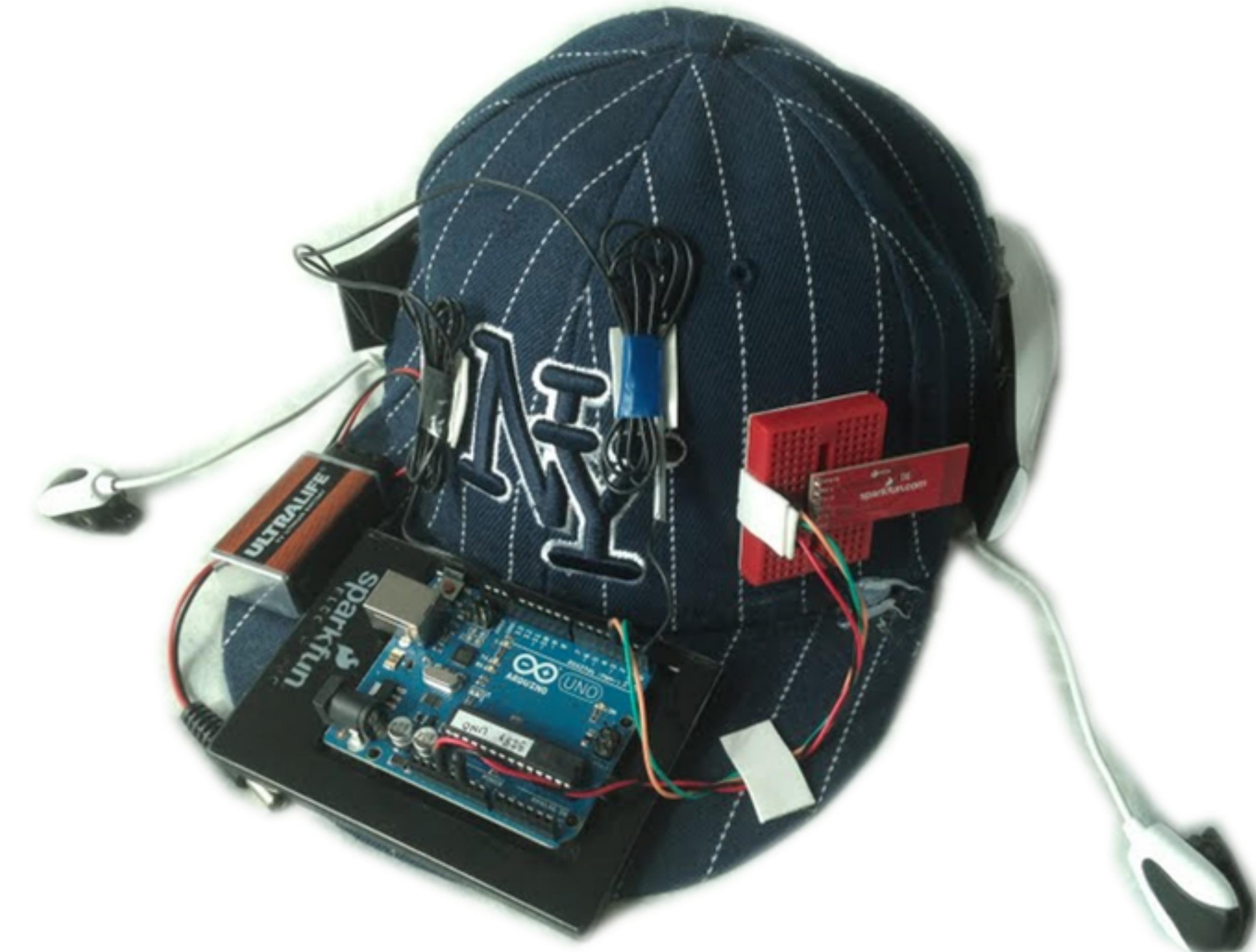
Imprimer son propre casque



Imprimer son propre casque



The first one



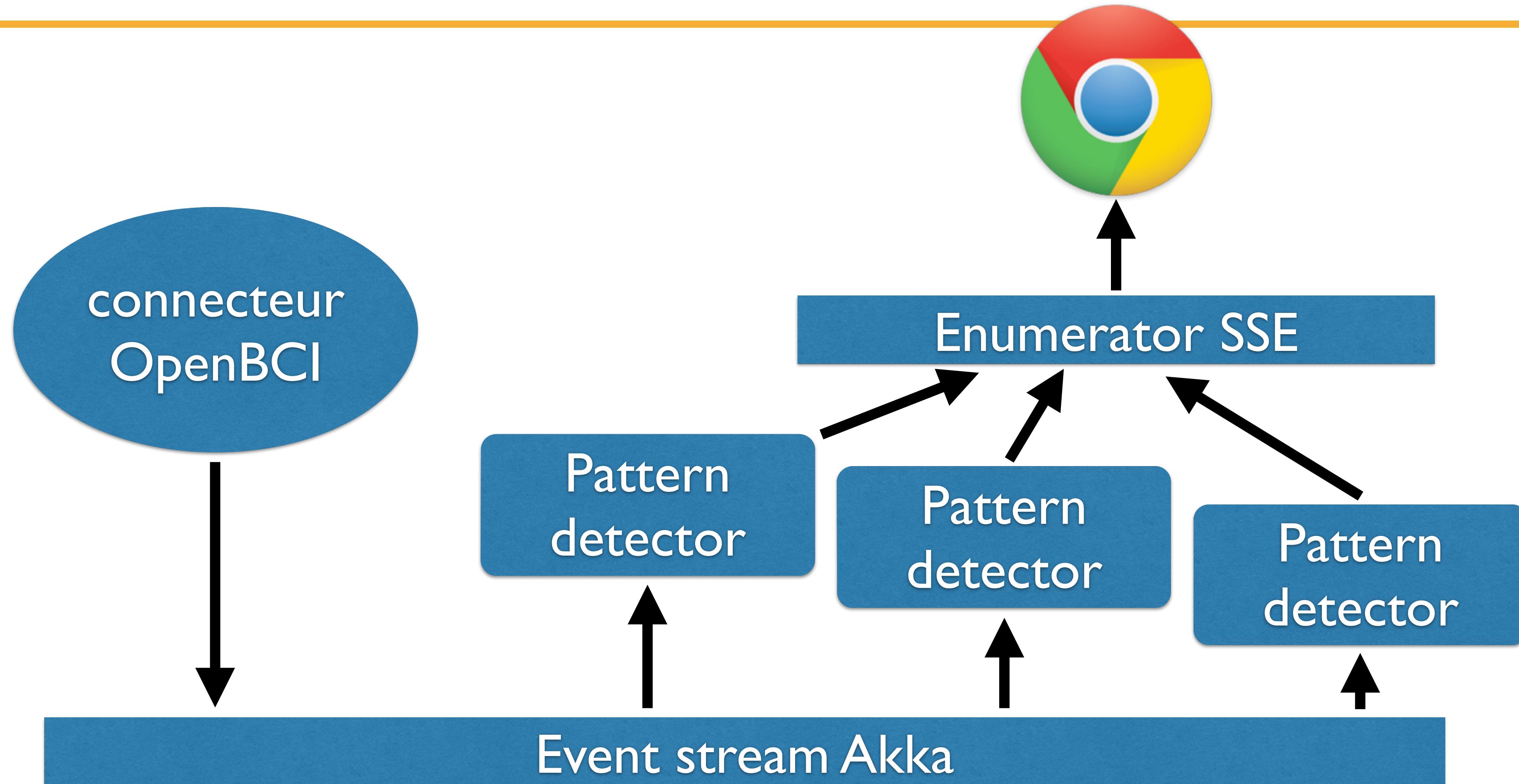
Demo



Comment ca marche ?

- Sur le serveur
 - Application Play 2
 - accès simple aux fonctionnalités web temps réel
 - Scala / Java
 - Connecteur à l'OpenBCI
 - stream les données dans un bus d'évènements Akka
 - Acteurs classifieurs / détecteurs
 - Acteur Akka abonné à une ligne de données en particulier
 - gère une série temporelle et une fenêtre glissante
 - Responsables de l'envoi de commandes vers le navigateur

Comment ca marche ?



Comment ca marche ?

- Dans le browser
 - Intégration des slides via pdf.js
 - API pour piloter le défilement, etc ...
- Utilisation intensive des Server Sent Event pour transmettre les « brainwaves » en temps réel
- Monitoring
- JQuery sparklines
- Demo
- Canvas et JavaScript

Problèmes rencontrés



Interfaçage avec l'EEG

- Pas d'API de haut niveau
- Pas de module Java
- Protocole basé sur une échange série
 - bien documenté
 - mais pas forcément simple à maîtriser
 - notamment au niveau de l'initialisation de l'EEG
- Création d'un librairie Java permettant de simplifier la communication
- <https://github.com/mathieuancelin/openbci-java>

OpenBCI GUI

```
//here is the processing routine called by the OpenBCI main program...update this with whatever you'd like to do
public void process(float[][][] data_newest_uV, //holds raw EEG data that is new since the last call
                     float[][] data_long_uV, //holds a longer piece of buffered EEG data, of same length as will be plotted on the screen
                     float[][] data_forDisplay_uV, //this data has been filtered and is ready for plotting on the screen
                     FFT[] fftData) { //holds the FFT (frequency spectrum) of the latest data

    //for example, you could loop over each EEG channel to do some sort of time-domain processing
    //using the sample values that have already been filtered, as will be plotted on the display
    float EEG_value_uV;
    for (int Ichan=0; Ichan < nchan; Ichan++) {
        //loop over each NEW sample
        int indexOfNewData = data_forDisplay_uV[Ichan].length - data_newest_uV[Ichan].length;
        for (int Isamp=indexOfNewData; Isamp < data_forDisplay_uV[Ichan].length; Isamp++) {
            EEG_value_uV = data_forDisplay_uV[Ichan][Isamp]; // again, this is from the filtered data that is ready for display

            //add your processing here...

            println("EEG_Processing_User: Ichan = " + Ichan + ", Isamp = " + Isamp + ", EEG Value = " + EEG_value_uV + " uV");
        }
    }
}
```

F
R
O
N
C
E

E
X
P
E
R
I
C
E

D
E
V
O
L
U
M
E

O
P
E
R
A
T
I
O
N

OpenBCI GUI

```
//here is the processing routine called by the OpenBCI main program...update this with whatever you'd like to do
public void process(float[][][] data_newest_uV, //holds raw EEG data that is new since the last call
                     float[][] data_long_uV, //holds a longer piece of buffered EEG data, of same length as will be plotted on the screen
                     float[][] data_forDisplay_uV, //this data has been filtered and is ready for plotting on the screen
                     FFT[] fftData) { //holds the FFT (frequency spectrum) of the latest data

    //for example, you could loop over each EEG channel to do some sort of time-domain processing
    //using the sample values that have already been filtered, as will be plotted on the display
    float EEG_value_uV;
    for (int Ichan=0; Ichan < nchan; Ichan++) {
        //loop over each NEW sample
        int indexOfNewData = data_forDisplay_uV[Ichan].length - data_newest_uV[Ichan].length;
        for (int Isamp=indexOfNewData; Isamp < data_forDisplay_uV[Ichan].length; Isamp++) {
            EEG_value_uV = data_forDisplay_uV[Ichan][Isamp]; // again, this is from the filtered data that is ready for display

            //add your processing here...
        }
        println("EEG_Processing_User: Ichan = " + Ichan + ", Isamp = " + Isamp + ", EEG Value = " + EEG_value_uV + " uV");
    }
}
```

F
R
O
N
C
E

E
V
O
X

D
O

OpenBCI GUI

```
public void mouseReleased() {  
    ...  
    for (int i = 0; i < nchan; i++) {  
        //was on/off clicked?  
        if (channelSettingButtons[i][0].isMouseHere() && channelSettingButtons[i][0].wasPressed == true) {  
            if (channelSettingValues[i][0] < maxValuesPerSetting[0]) {  
                channelSettingValues[i][0] = '1';  
                powerDownChannel(i);  
                deactivateChannel(i);  
            } else {  
                channelSettingValues[i][0] = '0';  
                powerUpChannel(i);  
                activateChannel(i);  
            }  
            writeChannelSettings(i); //write new ADS1299 channel row values to OpenBCI  
        }  
        ...  
    }
```

OpenBCI GUI

```
public void mouseReleased() {  
    ...  
    for (int i = 0; i < nchan; i++) {  
        //was on/off clicked?  
        if (channelSettingButtons[i][0].isMouseHere() && channelSettingButtons[i][0].wasPressed == true) {  
            if (channelSettingValues[i][0] < maxValuesPerSetting[0]) {  
                channelSettingValues[i][0] = '1';  
                powerDownChannel(i);  
                deactivateChannel(i);  
            } else {  
                channelSettingValues[i][0] = '0';  
                powerUpChannel(i);  
                activateChannel(i);  
            }  
            writeChannelSettings(i); //write new ADS1299 channel row values to OpenBCI  
        }  
    }  
    ...  
}
```

Le futur



Le futur

- Finir la librairie de connexion simplifiée avec l'EEG
- Ajouter un réseau de neurone
 - permet une classification plus aisée
 - plus adaptable
 - processus d'apprentissage pour chaque utilisateur
- Piloter un drone :-)



Q & A
