

# Modèle de rétine neuromimétique basé sur l'utilisation

## du projet OpenRetina

Pierre Albiges<sup>1</sup>, Laurent Perrinet<sup>2</sup>

<sup>1</sup>Aix Marseille Université, M1 NIC, Marseille, France <sup>2</sup>CNRS, INT, Marseille, France

### Motivations du projet

#### Biomimétisme

La biomimétisme consiste en la **reproduction de fonctions et de capacités des systèmes biologiques dans des systèmes artificiels**.

L'objectif est donc de s'inspirer de ce que la sélection naturelle a permis d'obtenir au cours de l'évolution pour résoudre des problèmes d'ingénierie limitant les systèmes artificiels actuels : résistance et adaptabilité aux contraintes environnementales, collecte et utilisation de l'énergie dans l'environnement, autonomie dans les déplacements et les prises de décisions, ...

L'ingénierie biomimétique est régulièrement utilisé dans le développement de technologies de pointe comme la robotique, mais aussi dans la recherche car il permet de mieux comprendre les mécanismes biologiques étudiés.

Dans le cadre de OpenRetina, c'est la **rétine et le système visuel animal** qui servent de modèle.

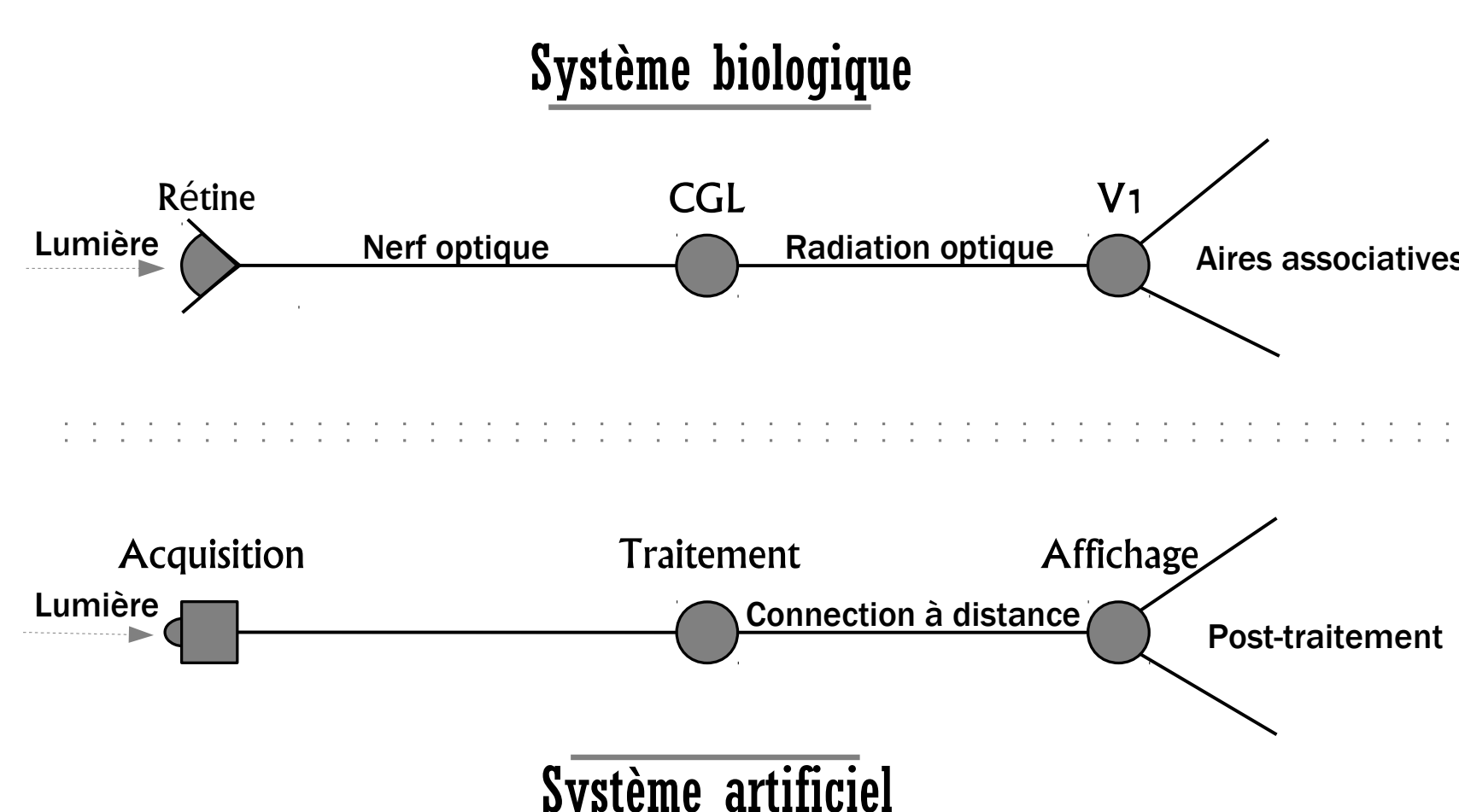


FIGURE 1 : Comparaison schématique du fonctionnement de OpenRetina et du système biologique équivalent

#### Low-tech

L'une des limites actuelles de l'ingénierie touche à la **puissance de calcul** des systèmes utilisés.

En opposition de l'ingénierie "high-tech", l'ingénierie "**low-tech**" se concentre sur l'intégration de programmes dans des systèmes artificiels possédant une **faible puissance de calcul** ou une **autonomie énergétique limitée** : systèmes embarqués (drones, systèmes robotisés autonomes) ou répondant à des contraintes particulières, tels que les rétines artificielles.

Dans le cadre de OpenRetina, c'est une **ingénierie low-tech et Open Source** qui est visée, afin de permettre sa diffusion et son utilisation par le plus de systèmes possible.



FIGURE 2 : Une Raspberry Pi 2 et sa picamera, principal support physique utilisé pour OpenRetina

### Méthodes computationnelles

OpenRetina est composé d'un ensemble de programmes majoritairement écrits dans le langage de programmation Python. Ces scripts réalisent une série de tâches qu'on peut diviser en catégories :

#### Grab

Récupérer une image ou une vidéo à partir d'un fichier video, d'une caméra locale ou d'une caméra distante (connectée à une Raspberry Pi).

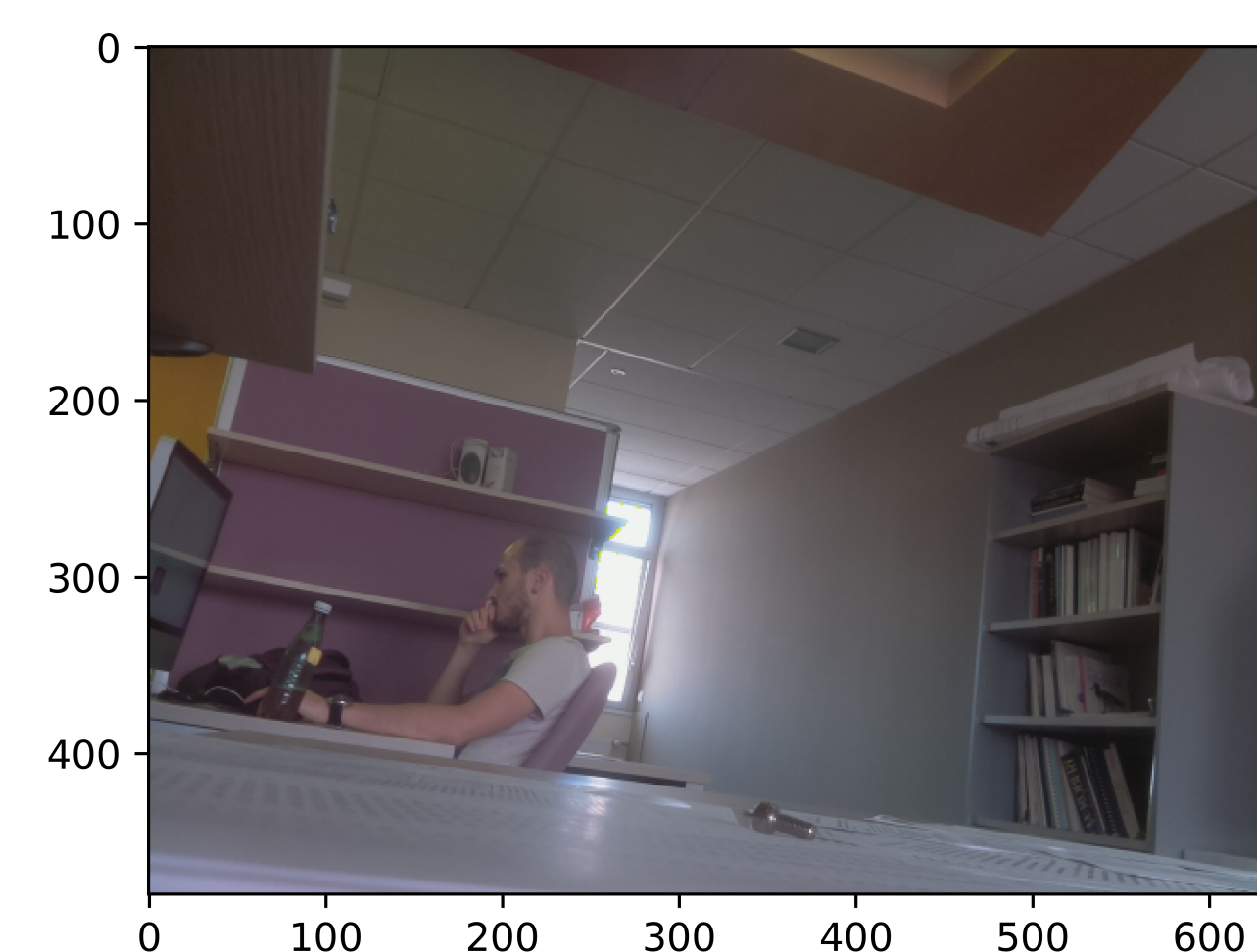


FIGURE 3 : Image brute capturée par une caméra distante, représentée sous forme d'un numpy array

#### Process

Réaliser un traitement sur chaque frame que comporte la capture, afin de **réduire la quantité d'informations transmises** et d'**accélérer la temporalité de la perception, sans perte de résolution**.

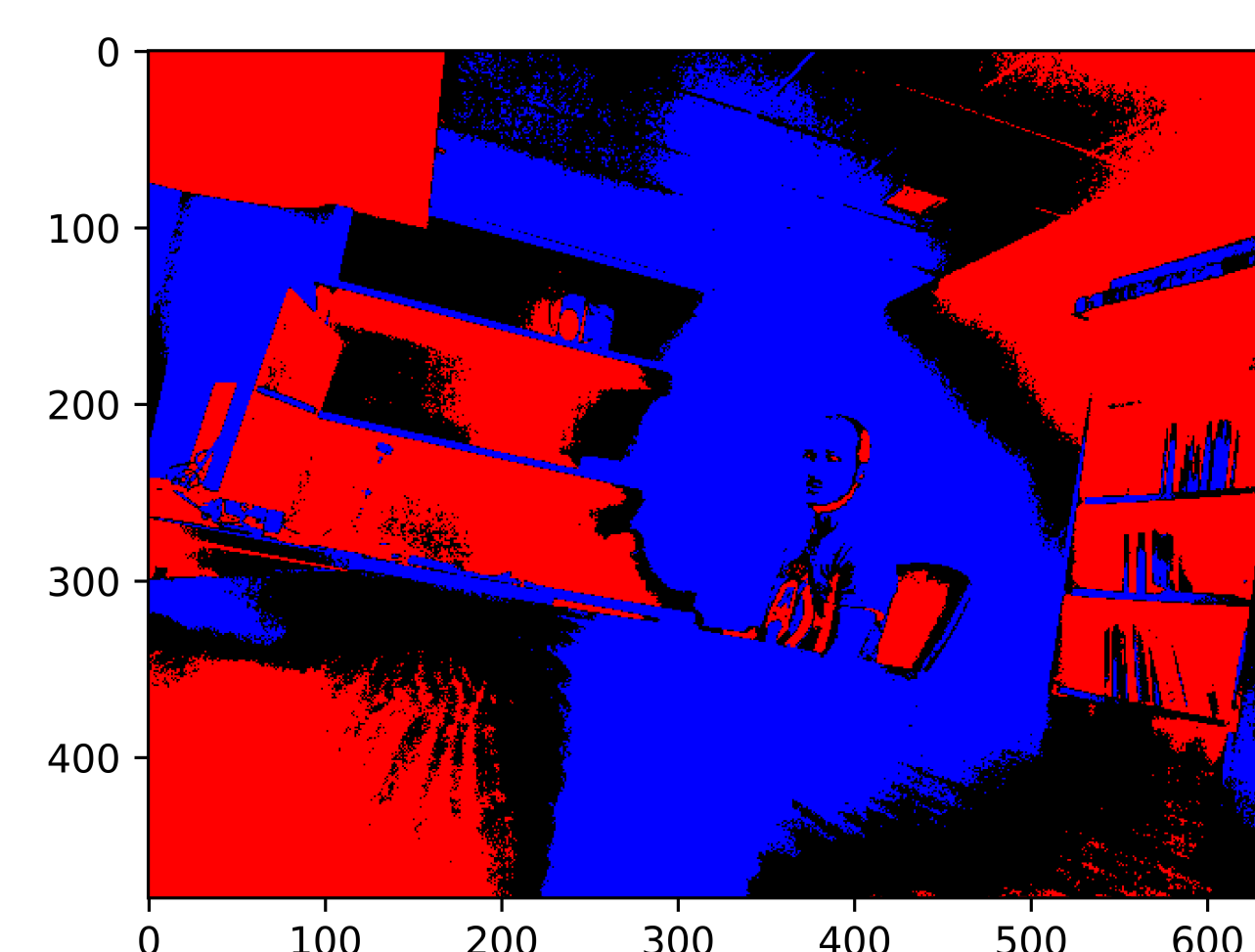


FIGURE 4 : Image après un traitement révélant les contrastes, représentée sous la forme d'un numpy array

#### Network

Transmettre les informations (requêtes ou données) entre les acteurs au travers un **protocole ssh** et l'utilisation de la **bibliothèque zeroMQ**.

#### Display

Afficher le résultat du traitement afin d'en contrôler les résultats.

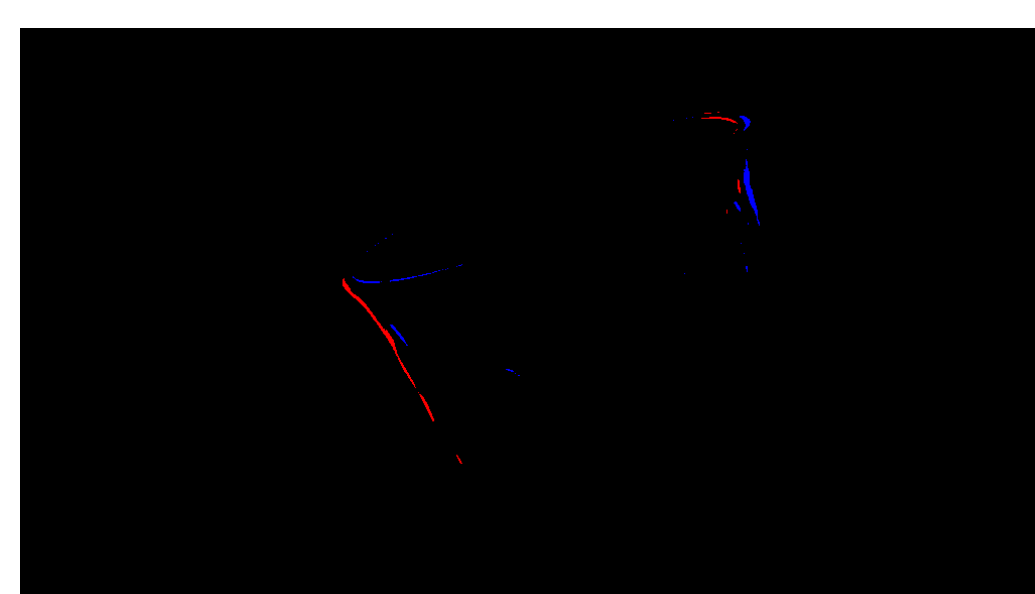


FIGURE 5 : Image après un traitement multi-couche ne révélant que les contours des objets en mouvement (extrait d'un affichage vidéo)

### Avancées apportées au projet

— Récupération des informations captées par la caméra pour reconstruire un fichier vidéo permettant de réaliser des **traitements hors-ligne et à postériori**

— Réalisation d'**étalonnages** pour contrôler l'influence de certains paramètres sur les performances du système

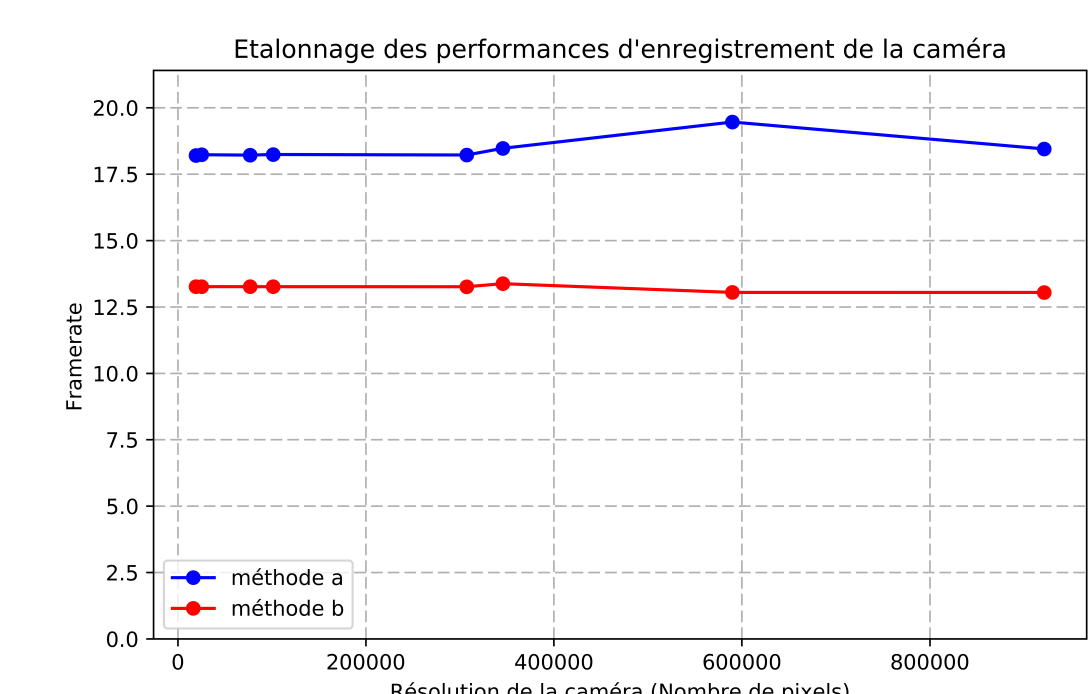


FIGURE 6 : Graphique issu de l'étalonnage comparant deux méthodes de calcul de la vitesse de capture

— Complexification du programme pour construire une **architecture en couches** : divise la charge de travail entre plusieurs acteurs, localise les erreurs et facilite la compréhension et les modifications

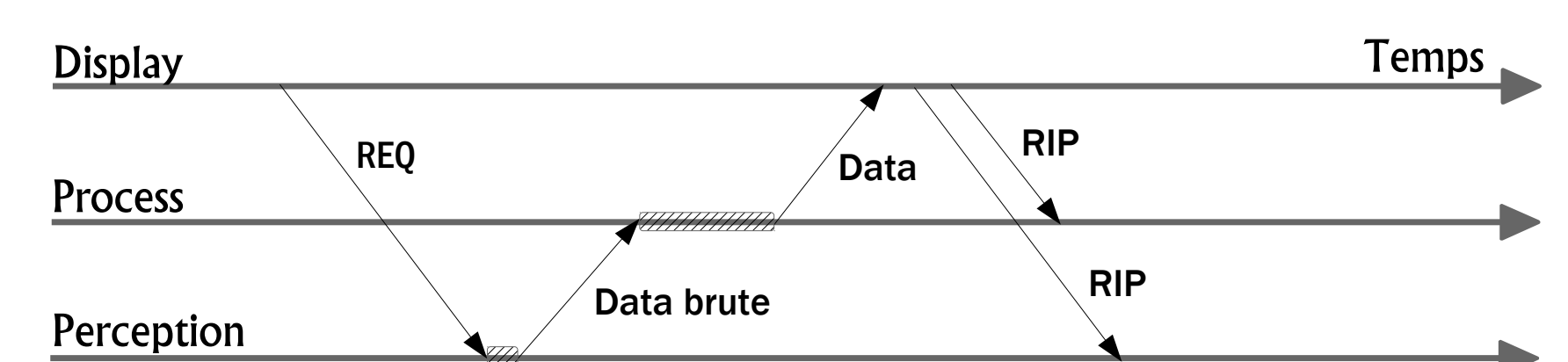


FIGURE 7 : Représentation schématique du fonctionnement dans le temps d'une architecture en 3 couches

### Compétences acquises

— Methodologie : approfondissement des **connaissances en Python** (notamment en programmation orientée objet) et en **TEX**, utilisation et maintien d'un **logbook**, utilisation de **git** pour gérer efficacement des versions, communications à distance via les **protocoles ssh** et **zeroMQ**

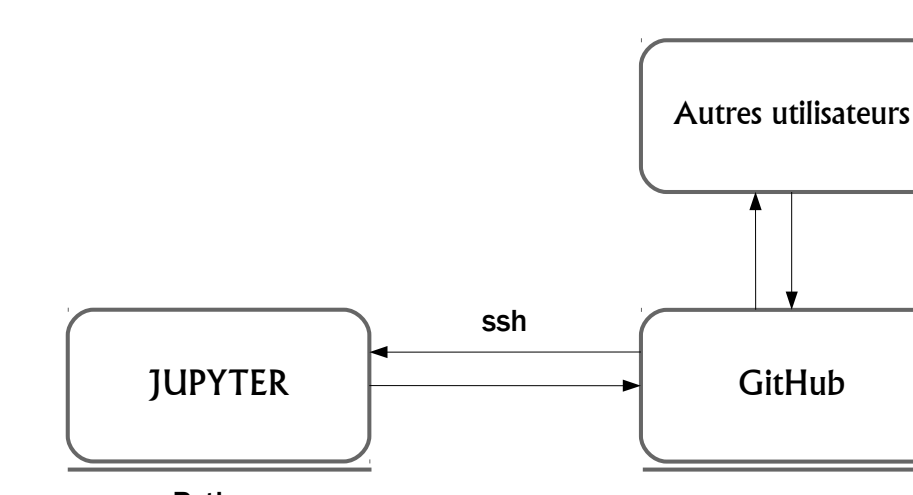


FIGURE 8 : Représentation schématique de la méthodologie utilisée pour OpenRetina

— Méthodes de traitement d'images et de vidéos

— Compétences d'ingénierie et de **reverse-engineering**.

— Connaissances neuroscientifiques dans les domaines de la perception et du traitement de l'information visuelle.