Année Universitaire
2015-2016

Institut Universitaire de Technologie
Aix-Marseille Université

LP Systèmes Informatiques et Logiciels
Killian NIOLLON

Dissection Virtuelle du Cou à partir d'acquisitions de scanner surfacique : un nouvel outil d'enseignement de l`anatomie et d'entrainement chirurgical

Stage en entreprise 14 semaines

Organisme d'accueil : Département Informatique :

Laboratoire d'Anatomie-Faculté de Médecine, Université de Montpellier site d'Arles

Laboratoire d'Informatique, de Robotique et Microélectronique de Montpellier (LIRMM)

Maitres de Stage :
Tuteur :
Mohamed Akkari Romain Raffin
Guillaume Captier

Gérard Subsol

Remerciements

J’ai réalisé mon stage au sein de deux organismes d’accueil : le Laboratoire d’Anatomie de la Faculté de Médecine et le Laboratoire d’Informatique de Robotique et Microélectronique de Montpellier, tous deux situés à Montpellier.

Je tiens à remercier tout particulièrement mes tuteurs : MM. Mohamed Akkari, Guillaume Captier et Gérard Subsol pour leurs investissements, conseils et la confiance qu’ils m’ont accordée tout au long du stage.

Je remercie également les personnels du LIRMM et du Laboratoire d’Anatomie pour m’avoir rapidement intégré dans leurs équipes, me permettant de réaliser mon stage dans les meilleures conditions.

Enfin, j’adresse mes remerciements à M. Chikaoui et Raffin ainsi qu’à ma famille pour leurs soutiens dans ma recherche de stage.

Résumé

J'ai effectué mon stage de fin de formation LP SIL au sein de l'équipe-projet de recherche ICAR du LIRMM situé à Montpellier. Chaque membre travaille sur un projet spécifique comme :
une plateforme d'annotations de vidéos, du Deep Learning, cryptage d'images, … .

Pour ma part, en collaboration avec la Faculté de Médecine, j'ai travaillé sur le projet suivant : « Dissection Virtuelle du cou à partir d'acquisitions de scanner surfacique».

Dans un premier temps, j'ai dû réaliser un outil de visualisation 4D (3D + temps) Web avec les interactions suivantes : Zoom (Avant / Arrière), changement des couches au clavier et rotation de l'objet 3D avec la souris.
Dans un second temps, il a fallu travailler sur le module « pédagogique » puis « évaluation ».
Le premier permet à l'étudiant d'apprendre l'anatomie et le second de vérifier ses connaissances en un temps limité.

Table Des Matières

Introduction

Dans le cadre de l'obtention de la Licence Professionnelle Systèmes Informatiques et Logiciels,
option Imagerie Numérique, un stage de 14 semaines doit être réalisé au terme de l'année de formation.
Ce stage a été effectué au LIRMM (Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier) en collaboration avec la Faculté de Médecine.
Il permet la mise en application des connaissances acquises à l'IUT en vue d'une insertion professionnelle après l'obtention du diplôme. Je tiens à préciser que le projet nommé « **Dissection virtuelle à partir d`acquisitions de scanner surfacique: un nouvel outil d'enseignement de l`Anatomie et d`entrainement chirurgical »** est financé par l'UNF3S.

Mon travail durant le stage consistait en la réalisation d'un outil de visualisation 4D dont l'aboutissement sera de permettre à un étudiant ou interne en médecine de réaliser une dissection complète virtuellement.
Ce projet répond à deux problèmes majeurs : le nombre d'étudiant en médecine en constante augmentation et le manque de cadavres disponibles pour l'entrainement à la dissection.
Dans cette application Web codée avec PHP, HTML 5, THREE.js, JavaScript et CSS 3 j'ai concentré mon travail sur l'élaboration d'un outil de visualisation 4D implémenté d'un module pédagogique et d'évaluation.

Au-delà des objectifs fixés avec l'entreprise, ce stage m'a permis d'atteindre mes objectifs personnels tels que : l'acquisition de nouvelles connaissances, l'immersion dans un milieu professionnel et gagner en assurance lors de mes exposés oraux.

Tout d'abord, je présenterai les organismes et services ou j'ai travaillé, puis j'expliquerai mes démarches pour réaliser l'outil de visualisation 4D. Enfin j'exposerai mon travail sur le module pédagogique et d'évaluation.

Mots-clés : **Objet 3D, Visualisation, Interactions et WEB.**

I Présentation des Organismes et du Service

1) Présentation de l'organisme : Faculté de Médecine de Montpellier, Laboratoire d'Antomie

Créée au XIIe siècle, la Faculté de médecine de Montpellier est la plus ancienne Faculté de médecine en exercice au monde. Rabelais, Lapeyronie, Chaptal, Arnaud de Villeneuve, Gui de Chauliac ont marqué son histoire.

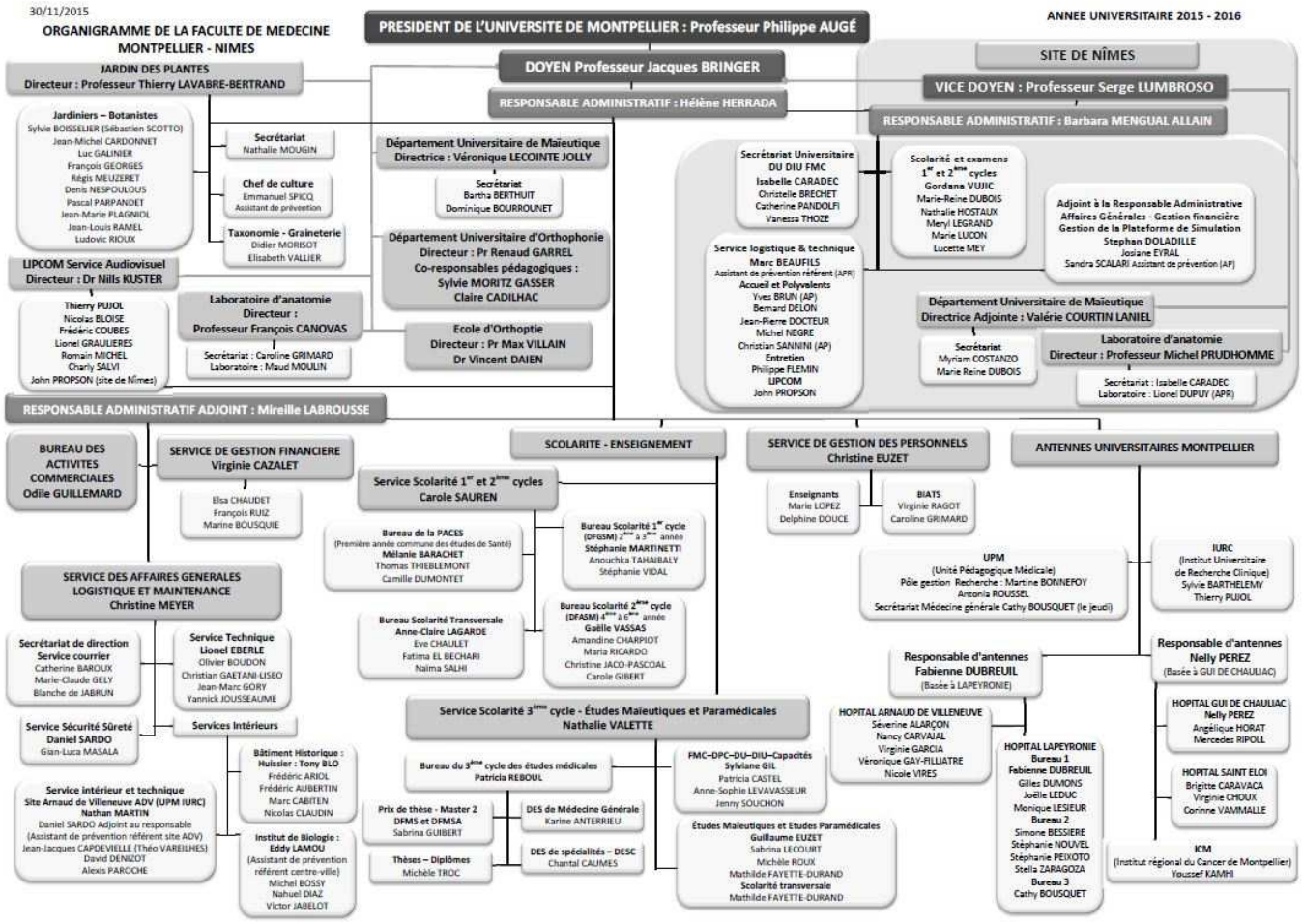
Les formations proposées par la Faculté de Médecine conduisent à différents diplômes :

- Diplôme de 3ème cycle
- Diplôme d'État de docteur en médecine
- Diplôme de Spécialités

Le Laboratoire d'Anatomie de Montpellier est une structure d'excellence dédiée à l'Enseignement et la Recherche à travers l'étude du corps humain. Les premières dissections réalisées à Montpellier dès 1556, et l'existence du TheatrumAnatomicum, créé par Jean Antoine Chaptal en 1795 et encore utilisé de nos jours, témoignent de son histoire ancienne et prestigieuse.

Sa mission principale est l'enseignement de l'Anatomie du corps humain à l'ensemble des étudiants en Médecine de la Faculté de Montpellier, mais également aux étudiants en Dentaire, Kinésithérapie et Maïeutique. Il constitue également un lieu de formation privilégié pour les futurs chirurgiens, toutes spécialités confondues.

Organigramme :



2. Présentation de l'organisme : le LIRMM

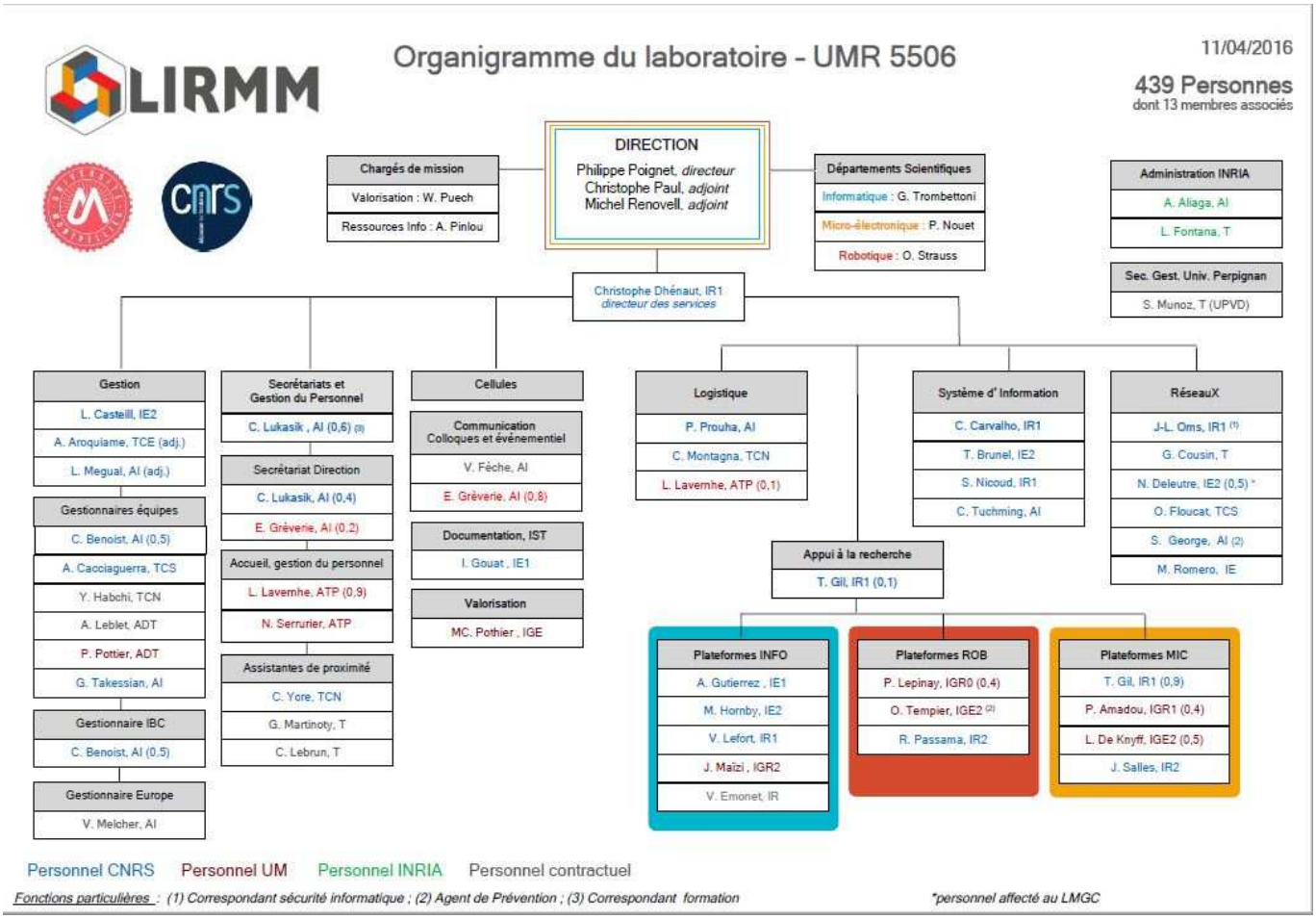
Le LIRMM est un laboratoire de recherche pluridisciplinaire dépendant de l'Université de Montpellier et du Centre National de la Recherche Scientifique (CNRS). Il est situé sur le Campus Saint-Priest de l'UM.
On trouve au sein de cette structure trois départements de recherche : informatique, microélectronique et robotique.

Département Informatique : Ici les thématiques de recherches s'étendent des mathématiques à la recherche appliquée au génie logiciel, aux applications de l'informatiques,…. .
Exemple : Bioinformatique, cryptographie, théorie des graphes,… .

Département Microélectronique : Les recherches sont centralisées autour de la conception et du test de systèmes intégrés et microsystèmes.

Département Robotique : Recherches sur la gestion de systèmes dynamiques complexes.
Exemple : Robot, pilotage de véhicules autonomes, traitement des images,…..

Organigramme :



3. Présentation du Service : Equipe-projet ICAR

L'équipe-projet ICAR (Image et Interaction) travaille au sein du département Informatique du LIRMM.
Leurs recherches associent interaction et traitement des données visuelles.
Exemple : images, vidéos et objets 3D.

Elle est structurée autour de trois axes majeurs : Analyse &Traitement (AT), Codage&Protection(CP) et Modélisation & Visualisation (MV).

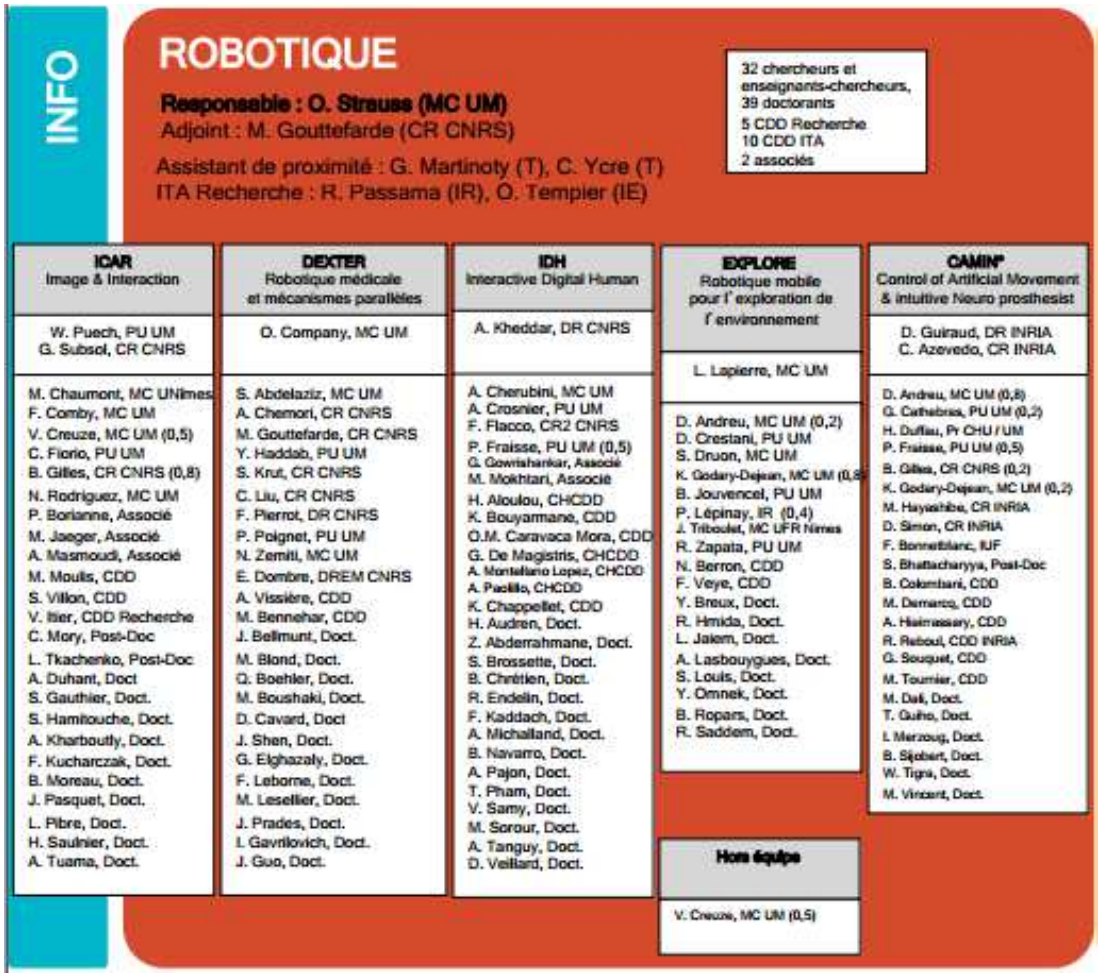
Axe AT : Etude des nouvelles techniques de traitement bas-niveau de l'information représentant : l'imprécis, l'incertain ou l'incomplet (types d'erreur en traitement des données).

Axe CP : On cherche ici à transmettre et archiver les données en toute sécurité.

On utilise le tatouage, la stéganographie ou le chiffrement pour y arriver.

Axe MV : Modélisation de grands ensembles de données pour les visualiser ou manipuler afin d'en extraire des connaissances.

Organigramme :



II SUJET DU STAGE

Le projet initié par le Laboratoire d'Anatomie est de développer un outil de dissection virtuelle dynamique (voir le projet UNF3S en annexe). Le principe est de superposer des acquisitions de scanner surfacique réalisées à différents instants sur différents niveaux de dissection pour créer un environnement quadridimensionnel (3D + temps) précis et réaliste.L'objectif est de former les étudiants et internes en médecine à l'anatomie et aux techniques de dissection.

Cahier des charges :

-Visualisation 3D+t : Proposer une plate-forme de visualisation 3D+t des maillages 3D acquis par scanner surfacique. La première étape consistera donc à réaligner les

maillages et uniformiser leurs géométries (découpage des parties non communes) et leurs couleurs.

-**Dispositif pédagogique** : Programmer un outil d'apprentissage de l'anatomie. Celui-ci pourra être très classique (points de repère annotés) ou plus original (Serious Game).

.

III Première Partie du Stage : Outil Visualisation 4D

1) Le Domaine Médical

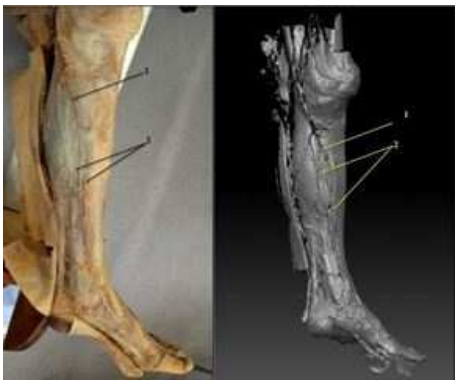
a) Etat de l'existant

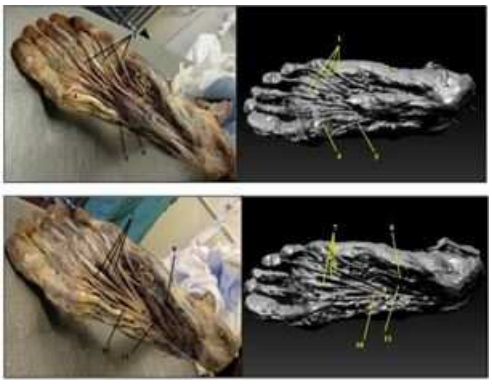
Avant de me lancer dans la réalisation du projet, j'ai réalisé une étude bibliographique et constaté que des études ont déjà été menées sur ce sujet.

-Travaux d'anatomie Sectionnelle (Visual Human Project, Chinese Visual Human, Korean Visual Human) : *Spitzer VM, Whitlock DG. The Visible Human Dataset: the anatomical platform for human simulation.Anat Rec. 1998 Apr;253(2):49-57.*
Zhang SX, Heng PA, Liu ZJ et al.The Chinese Visible Human (CVH) datasets incorporate technical and imaging advances on earlier digital humans.. J Anat. 2004 Mar;204(Pt 3):165-73.
Park JS, Chung MS, Hwang SB, Shin BS, Park HS.Visible Korean Human: its techniques and applications.Clin Anat. 2006 Apr;19(3):216-24

- Nécessité d'une segmentation manuelle laborieuse et chronophage.

-Welsh et Al (2014) : Utilisation du scanner surfacique pour recueillir la forme et l'apparence
(Couleur, texture,…) d'un objet.





Ces travaux sont soumis aux limites suivantes : cadavre formolé, absence de couleurs (nécessité de coupler avec photos). Enfin les acquisitions sont réalisés en fin de dissection, on ne se rend pas compte des étapes, ni des rapports entre chaque niveau de profondeur.

Solution : Utilisation du Scanner laser 3D de surface: Artec Spider (Société IMA solution de Toulouse). Avec celui-ci, on peut récupérer les textures et couleurs mais aussi faire les acquisitions durant les différentes étapes de dissection.

b) Formation sur l'anatomie du cou

J'ai dû découvrir un monde totalement inconnu à mes yeux, celui de la **Médecine.**
Pour réaliser un module « pédagogique » valable, les connaissances en informatique ne sont pas suffisantes. En effet, l'acquisition des fondamentaux sur les structures du cou (vasculaire, musculaire,…) est indispensable. Lors de mon stage j'ai appris à reconnaitre et localiser celles d'intérêt définies par mes maîtres de stage (une quinzaine) comme la veine jugulaire interne, les glandes sous mandibulaires, le muscle sternocléidomastoïdien… .

La formation s'est déroulée en deux étapes :

1- Phase théorique avec mes tuteurs (Mr Akkari et Captier) à l'aide de diaporamas et d'un atlas.

2- Phase pratique au côté de Mme Maud Moulin, thanatopracteur (préparatrice de corps) avec qui j'ai suivi et participé à la préparation d'un cadavre (ce qui consiste à rompre la rigidité cadavérique, nettoyer le corps, injecter du chlorure de zinc dans l'artère carotide pour le conserver.

« Cadavre en vue dorsale après injection du chlorure de zinc »

2) Le Domaine Informatique

a) Analyse de l'existant

A l'heure actuelle, l'utilisation de l'informatique dans le milieu médical s'améliore.

Néanmoins, dans le domaine de l'enseignement médical, le support non informatisé reste privilégié (schémas, dessins sur tableau,….).

Un des 'objectifs pédagogiques des enseignants en Anatomie est donc de moderniser les techniques d'apprentissage afin d'améliorer leur efficacité (visualisation tridimensionnelle) et leurs interactivité et leur disponibilité (car la dissection sur cadavre reste limitée par le petit nombre de corps donnés à la Science).De ce fait, des entreprises se sont déjà intéressées à ce sujet en concevant des applications web.

Exemple :

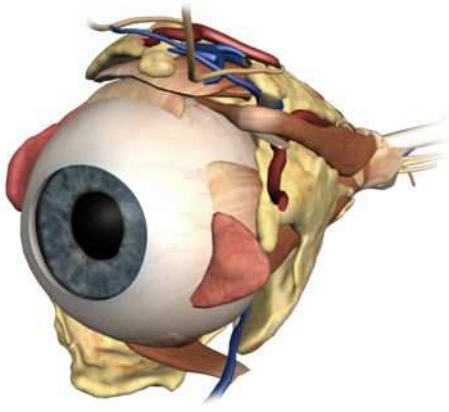
« Zygote Body» initié par Google puis repris par Zygote suite à la fermeture de Google Labs. Site web : https://zygotebody.com/.

Avec cette application, il est possible d'explorer le corps humain en 3D. Elle permet de visionner les différents éléments de notre anatomie grâce à un système de calques qu'il est possible d'afficher ou de faire disparaître à l'aide de curseurs qui gèrent leur transparence. On peut ainsi visualiser la peau, les muscles, les organes ou le squelette par exemple.

Limites:

-Les modèles visualisables sont « parfaits » et ne reflètent pas la réalité.

-L'accès à toutes les fonctions nécessitent un abonnement de 4$/mois.



« Humain ZygoteBody » « Œil Zygote Body »

Voici à présent plus en détail les outils et les données dont je disposais.

b) Choix des Outils et présentation des Données

b.1) Choix des Outils

Avant de commencer à coder, j'ai été confronté au choix suivant :

-Réaliser un exécutable indépendant qui nécessite l'utilisation de bibliothèque (openGL) avec le problème de la compatibilité des différents OS et la nécessité d'avoir les données sur l'ordinateur.

-Réaliser une application Web où les données seraient stockées sur un serveur distant.

Après une discussion avec mes tuteurs, j'ai décidé de concevoir une application Web pour pallier aux problèmes posés par l'exécutable indépendant.

Par conséquent j'ai installé un serveur web (WAMP) afin de pouvoir y stocker les données et lire le code php de mon application.

b.2) Les Données

J'ai travaillé sur des maillages 3D texturés possédant environ 1 million de faces. Les textures sont stockées dans des fichier bmp (bitmap) d'environ 50 000 ko.

c) Les Langages

HTML 5 (HyperText Markup Language) : C'est un langage utilisant des balises pour gérer et organiser le contenu (du texte, des liens, des images… [Pour afficher une image : <img src = 'chemin du fichier'></img>]).

CSS 3 (Cascading Style Sheets) : Il permet de gérer l'apparence de la page Web (décoration, couleur,….). C'est un complément du HTML.

PHP (Personal Home Page Tools) : C'est un langage de script que les serveurs interprètent à la volée et qui permet de rendre les sites dynamiques. Il permet l'actualisation de ceux-ci sans reécriture du contenu par le webmaster.

JavaScript : C'est un langage de script exécuté coté client (sur le navigateur de l'utilisateur). Il permet de réaliser des évènements lors d'une action de l'utilisateur (clic souris,…). Avec l'aide de Three.js, on peut réaliser des applications 3D avec une plus grande facilité.

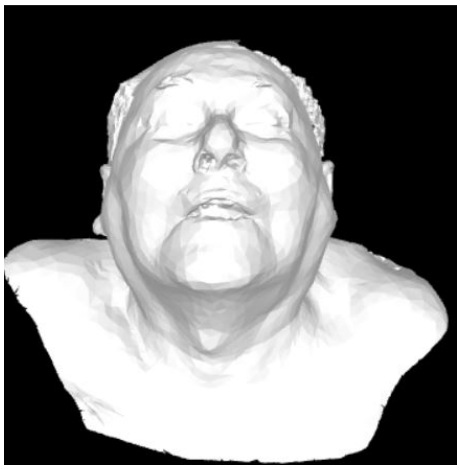
THREE.js : ***Three.js*** est une bibliothèque JavaScript pour créer des scènes 3D dans un navigateur web. Elle peut être utilisée avec la balise canvas du HTML5 sans avoir besoin d'un plugin. Pour l'intégrer à nos applications : <script type="text/javascript src="https://cdnjs.cloudflare.com/ajax/libs/three.js/r75/three.js"></script>

3) Réalisation

a) Le chargement des objets 3D

Cela a été l'étape la plus difficile dans la réalisation de **l'outil de visualisation 3D+t.**

1ere idée : La bibliothèque « xtk » (X Tool kit [https://github.com/xtk/X#readme]). Elle permet de réaliser des outils de visualisation scientifique, en particulier de données 3D. En apparence, elle a tout pour plaire. En effet, de nombreuses fonctionnalités sont intégrées (zoom et rotation de l'objet, chargement très simplifié des modèles 3D,…). Le problème, c'est qu'elle ne possède pas de lecteur MTL (Material Template Library). En résumé, on récupère la géométrie (normales, sommets, faces,…) de l'objet grâce au lecteur OBJ (Objet 3D) sauf que le fichier MTL qui contient les textures et les éclairages lui n'est pas lu. On obtient donc à l'écran un objet 3D sans texture.



« Objet 3D sans texture obtenu avec la bibliothèque xtk »

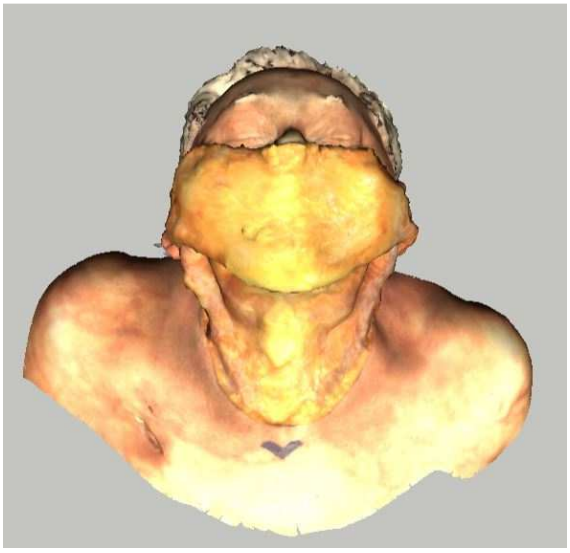
2eme idée : Lors de mes recherches sur THREE.js, j'ai remarqué l'intégration de plusieurs loader, notamment «ObjLoader » et « MtlLoader » (Lecteurs de fichiers Obj et MTL). Après l'ajout des fichiers .js à ma page contenant le corps de mon application, j'ai pu enfin les utiliser.

Remarque : xtk et three.js sont situées au même niveau (toutes les deux permettent de créer des scènes 3D dans un navigateur web). Le développeur doit cependant faire un choix car il ne peut pas combiner les deux bibliothèques dans une même application.

Voici un morceau de code commenté : « charger_objet_3d.js »

```
var onProgress = function ( xhr ) {
    if ( xhr.lengthComputable ) {
        var percentComplete = xhr.loaded / xhr.total * 100;
        console.log( Math.round(percentComplete, 2) + '% downloaded' ); // Permet suivre le chargement de l'objet
    }
};
var onError = function ( xhr ) { }; // Retour d'une erreur si l'objet n'est pas correctement chargé.
THREE.Loader.Handlers.add( /\.dds$/i, new THREE.DDSLoader() );
var mtlLoader = new THREE.MTLLoader(); //On instancie un Loader de fichier MTL
mtlLoader.setBaseUrl( 'planOriginal/' ); //On récupère URL du fichier
mtlLoader.setPath( 'planOriginal/' );// On récupère le chemin du fichier
mtlLoader.load( 'Plan_01_TexturesT.obj.mtl', function( materials ) {
    materials.preload(); //Lecture du fichier MTL et préchargement de son contenu dans la variable materials
    objLoader = new THREE.OBJLoader(); //On instancie un Loader de fichier OBJ
    objLoader.setMaterials( materials );// On applique le contenu du fichier MTL à l'objet 3D.
    objLoader.setPath( 'planOriginal/' );
    objLoader.load( 'Plan_01_TexturesT.obj', function (object1) {

        obj1 = object1;
        object1.visible = true; // L'objet est visible à l'écran
        object1.position.set(100,0,400); // Placement de l'objet par rapport à l'axe X,Y et Z
        object1.matrixAutoUpdate = false;


        scene.add(obj1);


    }, onProgress, onError );
});
```

Résultat :



« Objet Texturé avec THREE.js »

On obtient donc bien un objet texturé. J'ai chargé l'ensemble des maillages 3D avec la méthode présenté si dessus. Cependant ces derniers étant trop volumineux, l'espace mémoire du navigateur fut saturé. Pour résoudre le problème j'ai choisi le logiciel Meshlab. Avec celui-ci, j'ai utilisé la fonction Quadric Edge Collapse Decimation (with Texture) pour réduire le nombre de faces du maillage (de 1 000 000 à 10 000). Grâce à cela, la saturation de la mémoire du navigateur disparue et le problème fut résolu.

b) Le contrôle de la caméra : Rotation et Zoom

Pour commencer il faut créer une camera :

camera = new THREE.PerspectiveCamera (45, window.innerWidth / window.innerHeight, 0.1, 10000);

window.innerWidth : Récupère la largeur du contenu visible de la fenêtre de navigation.

window.innerHeight : Récupère la hauteur du contenu visible de la fenêtre de navigation.

Ensuite avec THREE.js, il est simple de réaliser des interactions avec la camera.

Démarche :

1-On crée un dossier « script » dans celui de notre projet.

2-On y ajoute le fichier « orbit_control.js ».

3-On ajoute <script src= 'script/orbit_control.js'></script> à la page contenant le script de notre application. (visualisateur.php).

4- On met un contrôle sur la caméra : **controls = new THREE.OrbitControls(camera);** à placer dans la fonction init() de notre application.

5- Dans la fonction animate() on ajoute controls.update() enfin de recalculer en permanence la position de la caméra.

Fonction animate() :

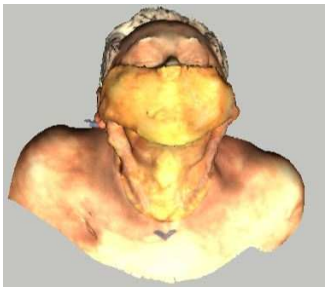
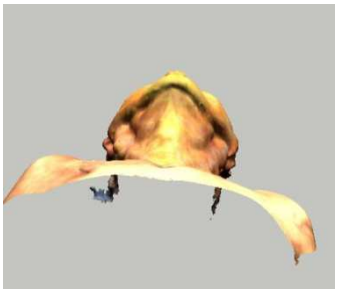
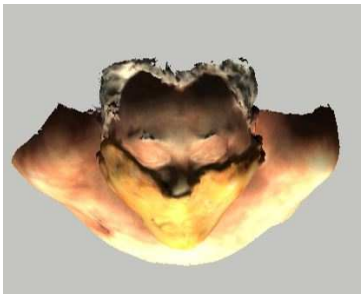
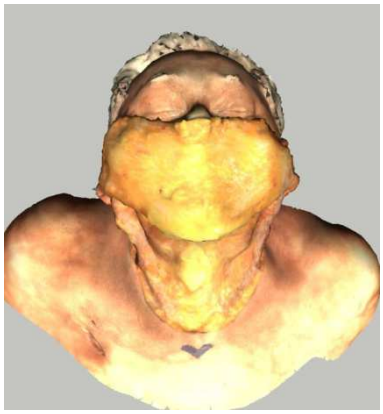
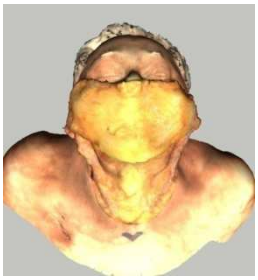
```
    function animate() {
            requestAnimationFrame( animate );
            controls.target = new THREE.Vector3(0, 0, 0);
            controls.update();
            render();
}
```

Fonction render() :

```
    function render() {
            camera.lookAt(scene.position);
            renderer.render( scene, camera);

}
```

Ces fonctions sont toutes les 2 indépendantes la fonction init().

Contrôles camera réalisables dans l'application :

<table>
<tr><td rowspan="3"> 
« Vue de face »</td><td> 
« Vue latérale droite »
Clic gauche enfoncé + bouger souris à droite</td><td> 
« Vue latérale gauche »
Clic gauche enfoncé + bouger souris à gauche</td></tr>
<tr><td> 
« Vue en contre plongée »
Clic gauche enfoncé +bouger souris vers le haut</td><td> 
« Vue en plongée »
Clic gauche enfoncé +bouger souris vers le bas</td></tr>
<tr><td> 
« Zoom Avant »
Molette souris vers l'avant</td><td> 
« Zoom Arrière »
Molette souris vers l'arrière</td></tr>
</table>

Au cours de mes tests, j'ai remarqué que l'objet 3D reste fixe, c'est la camera qui tourne autour de lui. Le problème est l'éclairage de l'objet qui reste identique quelque soit la position de la camera car la source de lumière est fixe et ne la suit pas.

c) Interaction Clavier : changement de plan

Basculer entre les différents plans de dissection est une fonctionnalité fondamentale de l'application. Dans celle-ci, cette action est réalisée au clavier.

Les étapes à suivre :

1- Mettre en place un gestionnaire d'évènement qui se déclenche lorsqu'une touche est enfoncée par l'utilisateur : window.addEventListener ('keydown', changerPlan, false);

2-Réaliser la fonction nommée changerPlan (event) ;

Un bout de code de la fonction : « visualisateur.php »

```
function  changerPlan(event){

    event = event || window.event; ///Permet l'utilisation de l'application par tous les Navigateurs

    if(event.keyCode == 80) ///Lorsqu'on appuie sur la lettre p. On avance dans les plans(1->2->3->...)
        {


            if(obj1.visible == 1)
                {
                    obj1.visible = false;
                    obj2.visible = true;
                    $('#plans').text('PLAN2');
                }

            else if(obj2.visible == 1)
                {
                    obj2.visible = false;
                    obj3.visible = true;
                    $('#plans').text('PLAN3');
                }
```

Au départ l'objet 1 est visible, lorsqu'on appuie sur la touche « p » celui-ci est occulté aux yeux de l'utilisateur puis l'objet 2 apparait,… . Cependant le premier est toujours présent en mémoire. En ce qui concerne $('#plans').text ('PLAN2'), il s'agit simplement d'un texte qui est modifié lorsque le plan change.

Seconde Partie du Stage : Les Modules

1) Le Module Pédagogique

a) L'interface Pédagogique

A ce stade du projet nous avions, un outil de visualisation 3D+t avec des interactions classiques.

Pour permettre l'apprentissage et la localisation des structures anatomiques, j'ai créé une interface pédagogique. En intégrant le fichier dat.gui.js, l'élaboration de celle-ci est facilitée.

Dat.gui.js : Script assistant le développeur dans l'élaboration d'interfaces graphiques.

Bout de code : « interface_pedagogique.js »

```
var gui = new dat.GUI( {width : 12 * 32 - 1});

/// Structures Musculaires
var MenuStructuresMusculaires = gui.addFolder('STRUCTURES MUSCULAIRES');

var MuscleSternocleomastoidien = { M_SternoCleidoMastoidien:function(){
    if(obj3.visible == 1 || obj4.visible == 1 || obj5.visible == 1){

        camera.position.set(-320, -80, 100);
        fleche_Muscle_Sterno_Cleido_Mastoidien.visible = true;
        sprite_Muscle_SternoCleidoMastoidien.visible = true;


    }

    else{
        alert("La structure selectionnée est présente dans le plan 3, 4 et 5 !");
    }

    }};
MenuStructuresMusculaires.add(MuscleSternocleomastoidien ,'M_SternoCleidoMastoidien');
```

var gui = new dat.GUI ({width : 12 * 32 – 1}) ;

➔ On crée une interface GUI et on précise sa largeur.

var MenuStructuresMusculaires = gui.addFolder('STRUCTURES MUSCULAIRES') ;

➔ On ajoute à l'interface Gui un sous-menu nommé STRUCTURES MUSCULAIRES.

var MuscleSternocleomastoidien = {M_SternoCleidoMastoidien : function (){…}} ;

➔ Permet de créer un bouton cliquable. Ici, si le Plan3, 4 ou 5 est visible à l'écran, on met en évidence le muscle sternocléidomastoïdien avec la camera puis on fait apparaitre la flèche et le sprite.

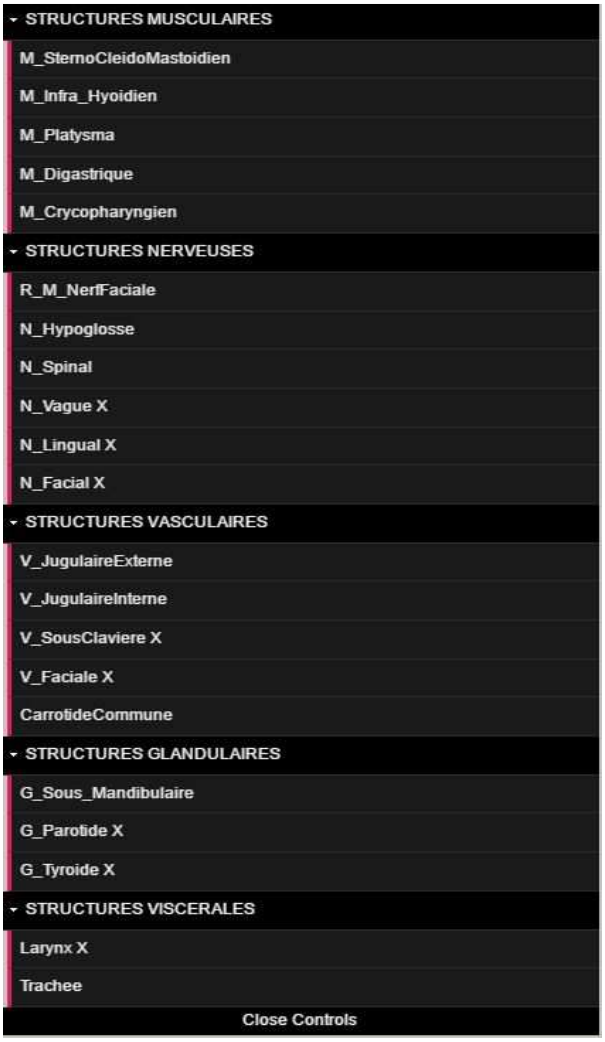
MenuStructuresMusculaires.add(MuscleSternocleomastoidien,'M_SternoCleidoMastoidin');

➔ On ajoute au sous-menu, le bouton dont le nom sera M_SternoCleidoMastoidien.

Résultat du code :

En suivant le même modèle, j'ai ajouté d'autres structures.

Au final on obtient :



« interface graphique obtenu avec dat.gui.js »

Les structures terminées par « X » signifient qu'elles ne sont pas visibles dans le maillage visualisé.

Lorsqu'on clique sur un des éléments des sous menus, on déplace la camera avec la fonction camera.position.set(x,y,z) afin de mettre en évidence la structure choisie par l'utilisateur. Cependant, pour faciliter d'avantage la visualisation, je les ai annotées sur l'outil de visualisation.

b) Annotation : Fléchage et Sprites

Afin d'optimiser la visualisation des structures, il m'a fallu annoter les objets 3D.

Au départ, j'ai commencé par flécher puis j'ai ajouté les sprites (petit bout d'image déplaçable) contenant leurs noms.

Le Fléchage :

Créer une flèche est aisée avec three.js, en effet elle contient une fonction prédéfinie : ArrowHelper (direction, Vecteur Origine, taille flèche, couleur).

Explication sur la création d'une flèche : « charger_fleche.js »

```
var hex = 0x0000ff; // couleur de la flèche (bleu)

///Fleches du Plan 2
var point1 = new THREE.Vector3(1, 1, 0);
var point2 = new THREE.Vector3(-1, 1, 1);
var direction = new THREE.Vector3().subVectors(point1, point2);
fleche_Muscle_Platysma_Coli = new THREE.ArrowHelper(direction.normalize(), point1, 100,hex);
fleche_Muscle_Platysma_Coli.position.set(-200,-10,100);
fleche_Muscle_Platysma_Coli.visible = false;
scene.add (fleche_Muscle_Platysma_Coli);
```

1- On créait deux vecteurs 3 dimensions avec des coordonnées qui leurs sont propres (points et 1 et 2).

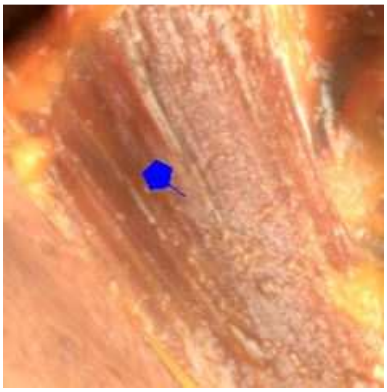
2-On créé une direction (ici, elle va du point1 vers le point2).

3-On instancie une flèche avec ArrowHelper(…). Ici, la direction est normalisée, elle a pour origine « point1 », une longueur de 100mm et une couleur bleue.

4-On positionne la flèche en fonction de X, Y et Z pour qu'elle pointe la structure désirée.

Remarque : Au départ la flèche est masquée, elle deviendra visible lorsque l'utilisateur cliquera sur le nom de la structure qu'il désire observer.

Muscle Sternocléidomastoïdien fléché :



Les Sprites :

L'intégration de cet élément est plus complexe. Il faut d'abord récupérer le fichier « sprite.js » sur internet. Une fois cela fait, il faut l'ajouter à notre application. Maintenant on peut instancier un sprite avec THREE.js

Explication sur la création d'un sprite : « charger_sprite.js »

```
var textureSprite_Muscle_Platysma =  new THREE.TextureLoader().load("structures/muscle_platysma.png");
var material_Sprite_Muscle_Platysma = new THREE.SpriteMaterial({map:textureSprite_Muscle_Platysma, color: 0x0000ff});
sprite_Muscle_Platysma = new THREE.Sprite(material_Sprite_Muscle_Platysma);
sprite_Muscle_Platysma.position.x = -200;
sprite_Muscle_Platysma.position.y = -10;
sprite_Muscle_Platysma.position.z = 100;
sprite_Muscle_Platysma.scale.set(25,10,1.0);
sprite_Muscle_Platysma.visible = false;
scene.add(sprite_Muscle_Platysma);
```

1- On charge une texture (Ici il s'agit de l'image muscle_Sus_Hyoidien.png).

2- On crée un matériau pour notre sprite dans lequel on applique la texture chargée précédemment.

3- On instancie un sprite dans lequel on applique le matériau que nous avons conçu.

4- On positionne le sprite en fonction des axes X, Y et Z à l'origine de la flèche.

5- On règle la dimension (largeur et longueur) du sprite.

6- On cache le sprite par défaut (il apparaitra quand l'utilisateur cliquera sur le bouton de la structure).

7- On l'ajoute à la scène.

Matériau : Variables contenant les paramètres propres à l'apparence de notre objet (texture, couleur,…).

Muscle Platysma (flécher + sprites) :

2) Module Evaluation :

a) Interface Evaluation

Elle reprend la forme de l'interface pédagogique, sauf qu'à la place du nom des structures, on y trouve des points « ? ». Lorsque l'utilisateur cliquera sur l'un des boutons du menu. Il devra identifier la structure anatomique concernée. Que sa réponse soit bonne ou mauvaise le « ? » sera transformé et contiendra la solution à la question.

Bout de code « interface_evaluation.js » :

```
var gui = new dat.GUI( {width : 12 * 32 - 1});

/// Structures Musculaires
 MenuStructuresMusculaires = gui.addFolder('STRUCTURES MUSCULAIRES');

 MuscleSternocleomastoidien = { M_SternoCleidoMastoidien:function(){

    if(obj3.visible == 1 || obj4.visible == 1 || obj5.visible == 1){
        valueM_Sterno.value = "1";
        camera.position.set(-320, -80, 100);
        fleche_Muscle_Sterno_Cleido_Mastoidien.visible = true;
        afficherDivFormStructMuscu();
        RepondreQcmStructureMusculaire(e);

}

    else{
        alert("La structure selectionnée est présente dans le plan 3, 4 et 5 !");
    }

    }};
itemMSterno =  MenuStructuresMusculaires.add(MuscleSternocleomastoidien ,'M_SternoCleidoMastoidien').name('?');
```

Le code est pratiquement identique à celui de l'interface pédagogique. On note tout de même quelques différences.

Ajouts :

AfficherDivFormStructMuscu()

➔ Permet d'afficher le QCM

RepondreQcmStructureMusculaire(e)

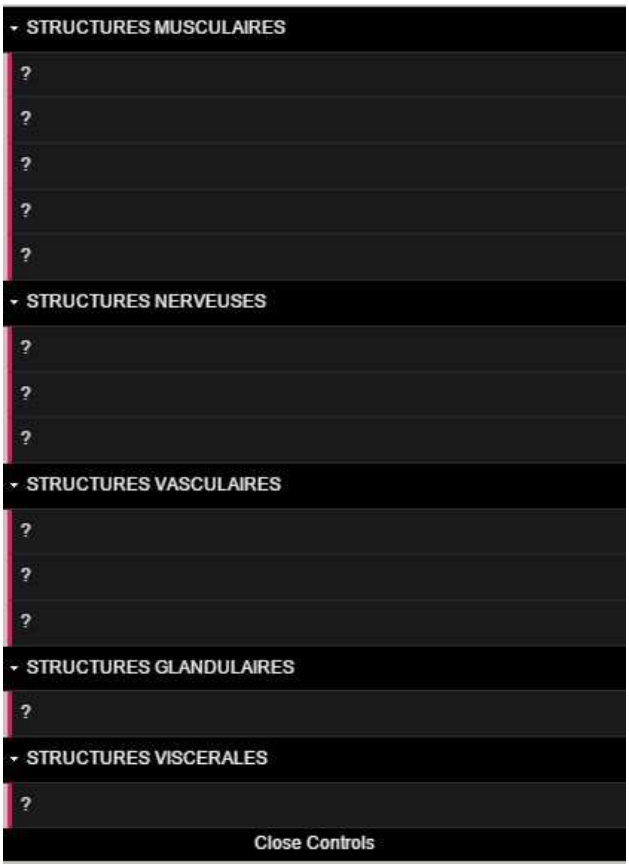
➔ Vérifie si la réponse de l'utilisateur est correcte ou non.

.name(' ?')

➔ Remplace le texte « M_SternoCleidoMastoidien » par « ? ».

Retrait : Les sprites ne sont pas présents lors de l'évaluation.

Au final on obtient :



Voyons à présent en détail le fonctionnement du QCM.

b) QCM

Voici l'étape qui m'a paru la plus complexe. En effet, il n'existe pas de scripts pour concevoir des QCM automatiquement. J'ai donc combiné HTML, CSS et JavaScript pour parvenir à le réaliser.

Je vais décomposer son élaboration en deux parties (HTML / CSS et JavaScript) pour faciliter la compréhension.

1- Partie HTML / CSS :

Le corps des QCM est codé en HTML puis pour la mise en forme on utilise le CSS.

Voici un exemple :

```
<div id="DivFormStructMusculaire">
    <form method="POST" id="FormStructMusculaire">
    <p class="ParaFormStructs">Nommez cette structure :<br/></p>
    <input type="radio" name="m_sternoCleido" id="m_sternoCleido"value="0" class="radio"/>
    <label for = "m_sternoCleido">M-SternoCleidoMestoidien</label><br/><br/>
    <input type="radio" name="m_platysma" id="m_platysma" value ="0" class="radio"/>
    <label for = "m_platysma">M-Platysma</label><br/><br/>
    <input type="radio" name="m_crycopharyn" id="m_crycopharyn" value="0" class="radio" />
    <label for = "m_crycopharyn">M-CrycoPharyngien</label><br/><br/>
    <input type="radio" name="m_digastrique" id="m_digastrique" value="0" class="radio"/>
    <label for = "m_digastrique">M-Digastrique</label><br/><br/>
    <input type="radio" name="m_Infra_Hyoidien" id="m_Infra_Hyoidien" value="0" class="radio"/>
    <label for = "m_Infra_Hyoidienn">M-Infra-Hyoidien</label><br/><br/>
    <input type="submit" value="Envoyer" class="button" />
    <input type="reset"  value="Reset" class="button"/>
    <input type="button" value="Quit" id="Quit" class="button" onclick="EffacerQcm()"/>
    </form>

</div>
```

Il s'agit d'un formulaire classique, on y trouve le texte « Nommez cette structure » avec plusieurs choix de réponses pour l'utilisateur. Il contient également un bouton « Envoyer » pour soumettre sa réponse, un « Reset » pour décocher une réponse non

souhaité et enfin un « Quit » dans le cas où l'utilisateur n'arrive pas à identifier la structure (Il fait disparaitre le QCM et peut identifier une autre structure puis revenir sur celle qui l'a bloquée).

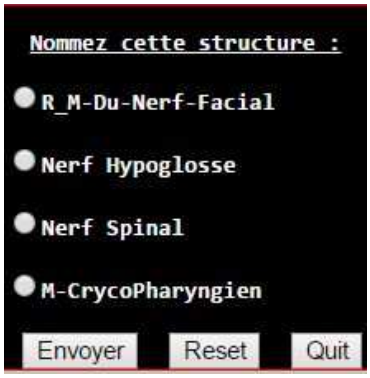
Pour la mise en forme, j'ai utilisé le code CSS suivant :

```css
#DivFormStructMusculaire
{
    width: 200px; /*largeur de la div */
    height: 235px; /*hauteur de la div */
    background-color: black;/*arrière plan de la div en noir*/
    border: 1px solid red; /*les bordures de la div seront rouges */
    font-weight: bold; /*Texte en Gras */
    display: none; /*partie importante: cacher la div et son contenu*/
    z-index: 9999; /*partie importante: à l'affichage, la div sera au-dessus de tout le reste*/
    position: absolute; /*positionnement par rapport au coin haut gauche de l'écran */
    right: 0px;
    bottom: 0px;
    margin-right: 10px;
    margin-bottom: 10px;

}
```

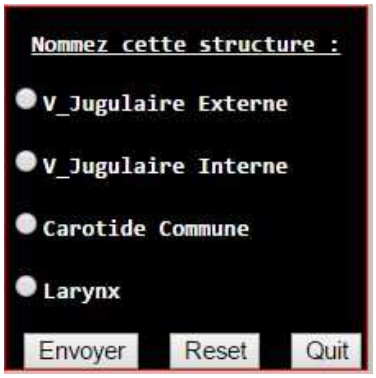
Voici tous les QCM de l'application :

QCM Structures Musculaires :

QCM Structures Nerveuses

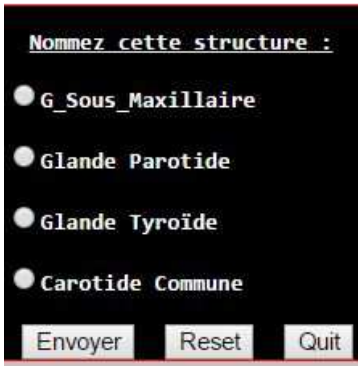


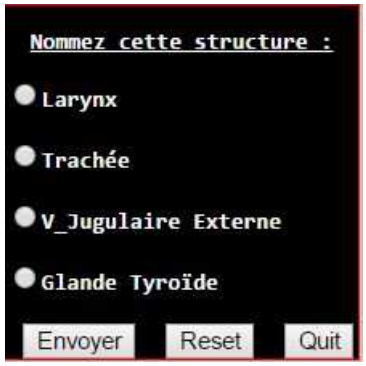
QCM Structures Vasculaires :



QCM Structures Glandulaires :

QCM Structures Viscérales :





2- Partie JavaScript :

Nous avons à présent tous nos QCM.

Pour rappel, dans « l'interface évaluation », j'ai évoqué l'ajout de nouvelles fonctions. **afficherDivFormStructMuscu()** et **RepondreQcmStructureMusculaire(e)** dans le cas où l'utilisateur clique sur l'une des structures musculaires.

Remarque : Les autres QCM possèdent 2 autres fonctions qui leurs sont propres. Exemple, Dans le cas où l'utilisateur sélectionne une structure nerveuse, il fera appel à **RepondreQcmStructureNerveuse(e)** et **afficherDivFromStructNerveuse(),….** .

Je présenterai ici uniquement les fonctions concernant les structures musculaires.

afficherDivFormStructMuscu() :

```
function afficherDivFormStructMuscu()
{
    document.getElementById('DivFormStructMusculaire').style.display = 'block';

}
```

Elle est très simple, on récupère l'identifiant du QCMstructureMusculaire pour modifier ces propriétés.

Ici on l'affiche à l'écran lorsque l'utilisateur clique sur une des structures musculaires.

RepondreQcmStructureMusculaire(e) [bout de code]:

```
function RepondreQcmStructureMusculaire(e)
{

    if(valueM_Sterno.value == 1)
    {

        if((FormStructMusculaire.m_sternoCleido.checked)==1)
            {
                alert("Bonne Réponse");
                resultat+=1;

            }
        else{alert("Mauvaise Réponse");}

    MenuStructuresMusculaires.remove(itemMSterno);
    itemMSterno = MenuStructuresMusculaires.add(MuscleSternocleomastoidien ,'M_SternoCleidoMastoidien');
    document.getElementById('m_sternoCleido').disabled = true;

    }
```

...

```
    compteur+=1;
    e.preventDefault();
    EffacerQcm();
    ReinitialiserValueForms();
    if(compteur==14){alert('Vous avez fini l\'Evaluation en ' + minu + ' min :' + secon + ' secondes');
                     clearTimeout(myTimer);
                     RendreBoutonsInactifs();
                     CalculerScoreFinal()};


}
```

Au départ l'attribut « value » de toutes les structures est initialisé à 0. Lorsque l'utilisateur clique sur le « ? » correspondant au muscle sternocléidomastoïdien dans l'interface évaluation sa « value » passe à 1. On vérifie que celui-ci a coché la bonne réponse. Si oui, on affiche « Bonne Réponse » et il gagne un point. Sinon, on affiche « Mauvaise Réponse ». Dans tous les cas, on supprime le « ? » que l'on remplace par le nom de la structure et on empêche l'utilisateur de recliquer sur la réponse exacte.

Le QCM est également soumis à une contrainte de temps (5 min). On note 2 scénarios qui mettent fin au QCM : Soit l'utilisateur a répondu à toutes les questions dans le temps imparti, soit il a dépassé les 5min. (Dans le second cas, toutes les réponses justes sont bien évidement comptabilisées dans la note finale).

Intéressons-nous enfin au calcul et stockage du résultat final.

c)Calcul et Stockage du Résultat Final

c .1) Partie Calculatoire :

CalculerScoreFinal() :

```
function CalculerScoreFinal(){

        score = resultat * 20 / 13;
        alert('Votre Note: ' + score + '/ 20');
        EnvoyerResultat();

}
```

Le score final répond à la formule suivante : (nombre bonne réponse * 20 / nombre de structures).

On affiche la note de l'utilisateur que l'on stocke dans un fichier texte dédié.

c.2) Partie Stockage :

Nous faisons face ici à un problème plus complexe, en effet il faut utiliser AJAX et PHP pour envoyer et traiter les données puis les stocker dans le fichier.

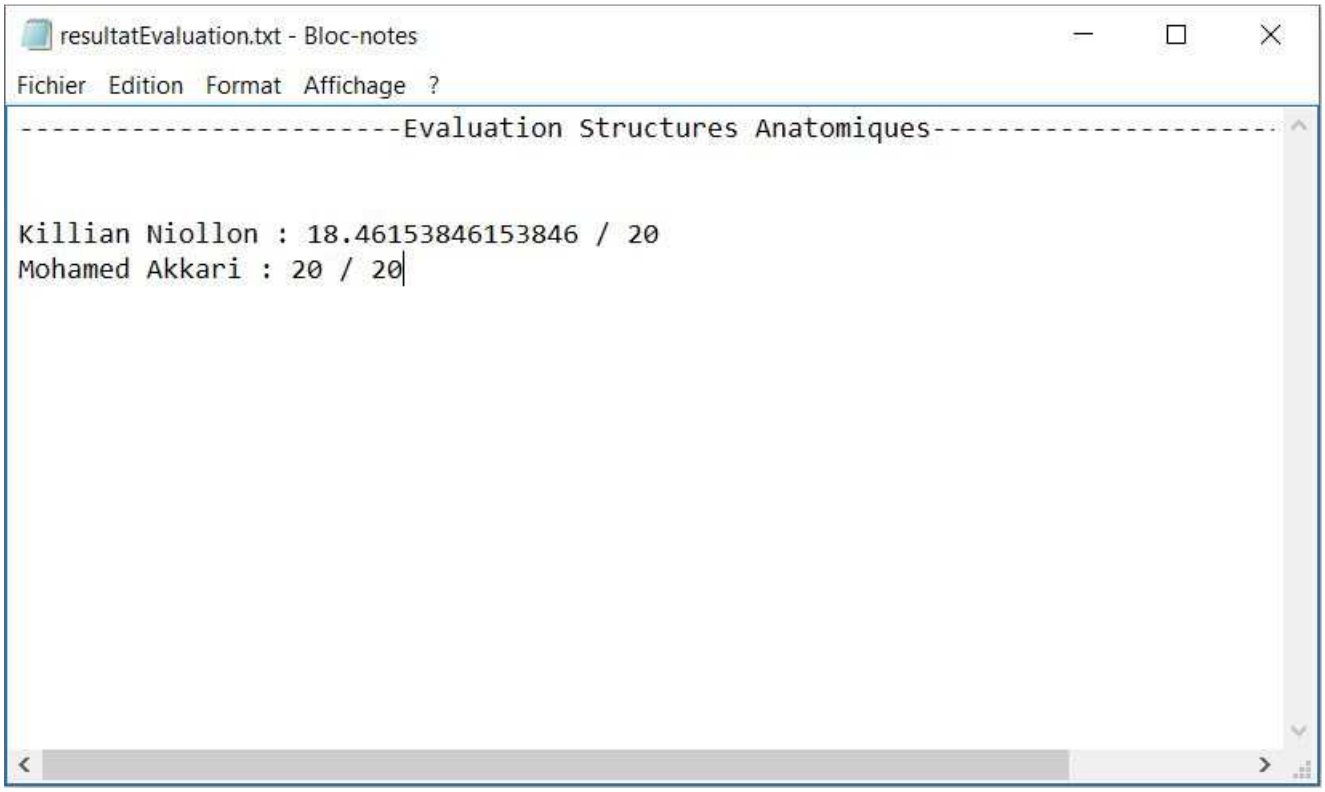
EnvoyerResultat() :

```
function EnvoyerResultat(){

    var xhr = getXMLHttpRequest();
        xhr.open("GET", "traitement.php?prenom=" + prenom + "&nom="+ nom +"&score="+ score, true);
        xhr.send(null);


}
```

L'envoie des données est réalisé grâce à une requête http. Ici, on transmet, le prénom, nom et score de l'utilisateur par la méthode GET (transmission des données dans l'url) à la page « traitement.php »

Traitement.php :

```
<?php

header("Content-Type: text/plain");

$prenom = $_GET['prenom'];
$nom = $_GET['nom'];
$score  = $_GET['score'];

if(isset($prenom)&&isset($nom)&&isset($score)) // Si les variables existent
{
    $monFichier = fopen('resultatEvaluation.txt', 'a'); /// Ouverture fichier resultatEvaluation.txt
    fputs($monFichier, $prenom); /// On écrit le prenom dans le fichier
    fputs($monFichier, ' ');
    fputs($monFichier, $nom); /// On écrit le nom dans le fichier
    fputs($monFichier, ' : ');
    fputs($monFichier, $score); /// On écrit le score dans le fichier
    fputs($monFichier, ' / 20');
    fputs($monFichier, "\r\n");
    fclose($monFichier);


}
else{}

?>
```



Conclusion

Cette immersion dans une entreprise durant 14 semaines m'a été des plus bénéfiques.

Sur le plan professionnel, elle m'a permis de développer grandement mes compétences.

Je suis à présent capable de réaliser des outils de visualisation 3D+t avec des interactions, ce dont je ne pensais pas être capable au début de mon stage. Le contact de personnes investies dans leur domaine m'a poussé à me surpasser pour progresser sans cesse.

Je suis ravi d'avoir pu découvrir le domaine médical qui m'a passionné.

En effet, malgré mon manque total de connaissances mes tuteurs m'ont accordé leur confiance et parfaitement accompagné afin que je puisse tirer le maximum de cette expérience. A noter que la préparation d'un cadavre restera un moment fort du stage.

Sur le plan humain, j'ai gagné en assurance grâce aux nombreuses réunions d'équipes et muri aux contacts de mes partenaires de travail.

Je tiens donc à remercier une nouvelle fois, le laboratoire d'anatomie, l'équipe-projet ICAR et mes tuteurs (M. Akkari, M.Captier et M.Subsol) pour m'avoir considéré comme un membre de leurs équipes à part entière.

Enfin, cette expérience m'a donné envie d'intégrer le milieu professionnel rapidement afin de mettre à profit mes nouvelles compétences au sein d'une entreprise idéalement traitant des projets alliant médecine et informatique.

ANNEXES

1) Technique :

a) Squelette d'une application Web

Elle doit contenir impérativement les éléments suivant :

Le renderer : Il s'agit d'une vue de l'application. Il affichera le contenu de la scène à l'utilisateur. **[renderer = new THREE.WebGLRenderer();]**

La scène : C'est le cœur de notre application, elle contient les objets, éclairage,… de notre application. **[scene = new THREE.Scene();]**

L'éclairage : Permet de visualiser les couleurs / textures des objets.

[ambient = new THREE.AmbientLight(0x444444);]

La camera : Rend notre application plus dynamique (rotation, zoom,…)

[camera = new THREE.PerspectiveCamera(45, window.innerWidth / window.innerHeight, 0.1, 10000);]

b) Manuel d'Utilisation

Touche « p » : Avancer dans la visualisation des plans de dissection : (1-2-3-4…-8).

Touche « s » : Reculer dans la visualisation des plans de dissection : (1-8-7-6-5…)

Touche « r » : Réinitialise la caméra dans une position prédéfinie.

Click Gauche enfoncée + déplacement souris : Rotation Objet.

Molette souris : Zoom Avant / Arrière.

d) Améliorations

- Rendre l'application « modulable », séparer les données du code.

C'est une réelle difficulté, il faut stocker les données dans des fichiers externes. De plus, il faut une grande maitrise du php et d'AJAX pour parvenir à les lire afin de générer toute l'application dynamiquement.

- Rotation de l'objet et non de la caméra.

Si c'est l'objet qui tourne, son éclairage changera, alors que pour l'instant il ne varie pas.

- Fléchage des structures à la souris.

-Contourage des structures.

-Règle graduée.

Permet à l'étudiant de mesurer les structures anatomiques.

-Niveaux de difficultés dans le module évaluation

2) Fiche de Temps

Du 14/03/16 au 18/03/16 : Découverte des locaux, tuteurs et projet.

Du 21/03/16 au 25/03/16 : Etude bibliothèque xtk (Récupération objet 3D + zoom/rotation).

Du 28/03/16 au 01/04/16 : Réalisation d'une nouvelle application avec three.js (chargement objets + récupération des textures).

Du 03/04/16 au 17/04/16 : Finalisation de l'outil de visualisation 4D + ajout menu des structures anatomiques.

Du 18/04/16 au 01/05/16 : Travail sur le module pédagogique (fléchage structures et sprites).

Du 02/05/16 au 17/05/16 : Travail sur le module évaluation (interface et QCM) et stockage du résultat.

Du 18/05/16 au 30/05/16 : Finalisation Page d'accueille et réalisation du rapport de stage.

Du 31/05/16 au 07/06/16 : Ajout du menu « accès rapide plan », rotation précise de la camera autour de l'objet avec le pavé directionnel.

3) Lexique

Rompre la rigidité cadavérique : Rendre le cadavre « souple » afin de la manipuler plus simplement.

Acquisition surfacique : Obtention des maillages 3D à l'aide d'un scanner pour les utiliser sur un ordinateur.

Deep Learning : Le « *deep learning* » fait partie d'une famille de méthodes d'apprentissage automatique fondées sur l'apprentissage de modèles de données. Une observation (exemple : une image) peut être représentée de différentes façons par un vecteur de données en fonction de :

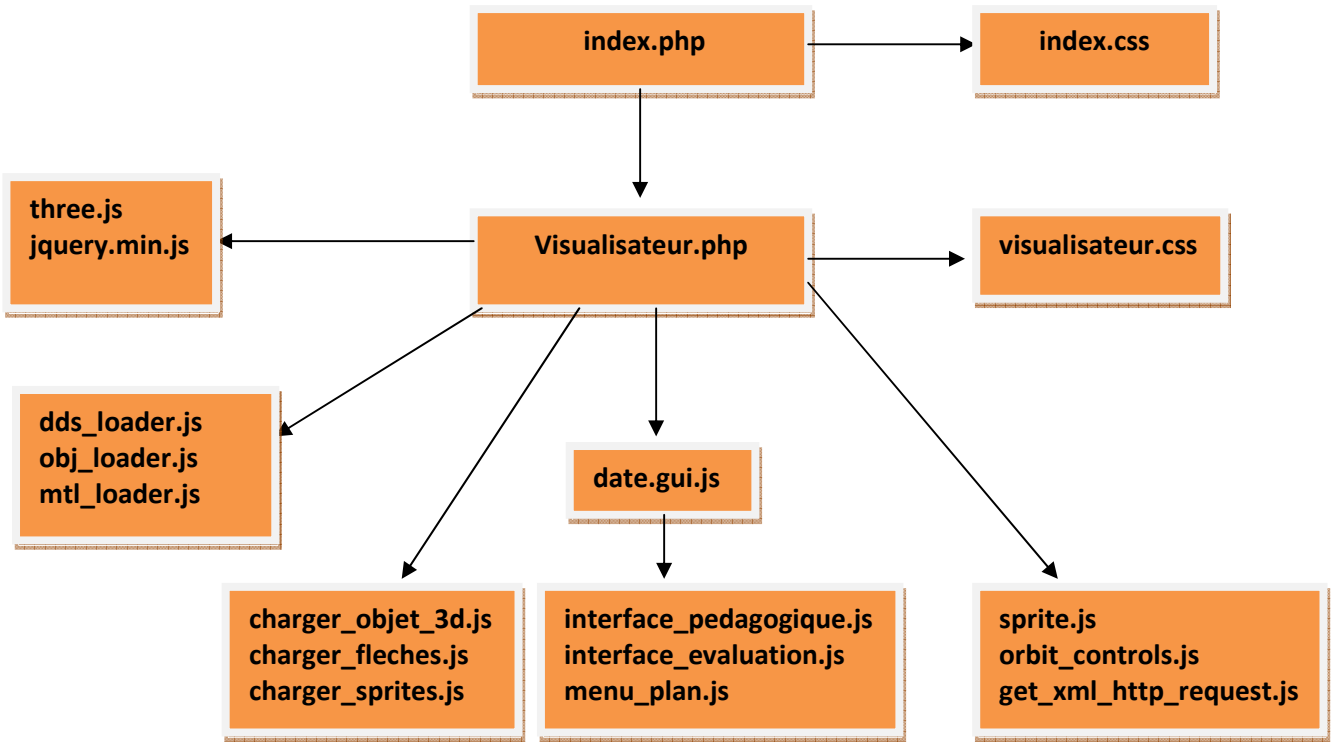
- l'intensité des pixels dont elle est constituée.
- ses différentes arêtes,
- les différentes régions de forme particulière,

- etc.

Cryptographie : Méthode permettant de protéger des messages à l'aide de secrets ou clés.

Stéganographie : Art de la communication secrète. L'objectif est de dissimuler un message secret dans un médium anodin (image, vidéo, son,…) de sorte qu'il ne puisse être détecté.

4) Structure de l'Application :



5) Fiche Projet

Projet M1603

FICHE PROJET

Appel à Projets : 2015-2016

Dissection virtuelle à partir d`acquisitions de scanner surfacique: un nouvel outil d'enseignement de l`Anatomie et d`entrainement chirurgical	
Université contractante	**Montpellier**
Date de dépôt du projet	15/09/2015
Financement demandé	10 000,00 €
Financement accordé	**6 600,00 €**
Thème(s)	Anatomie
Public(s)	
Partenaire(s) éventuel(s)	Laboratoire d` Informatique Robotique et Microelectronique de Montpellier (LIRMM) équipe ICAR, CNRS/Université de Montpellier
Durée du projet	10 mois
Heure enseignement numérique :	
- nombre d'heures	100
- coût par heure	100,00 €

Porteur de Projet

Nom, prénom	**AKKARI MOHAMED**
Fonction	Chef de Clinique Assistant en Anatomie
Résidence administrative	UFR Médecine Université de Montpellier Laboratoire d`Anatomie 2 rue École de médecine - CS 59001 34060 - Montpellier
Téléphone fixe	04 34 43 35 65
Téléphone mobile	06 88 18 24 92
Courriel	mohamed.akkari.orl@gmail.com
Gestionnaire convention / Assistant(e)	

Le Président du Comité des Projets UNF3S
Marcel SPECTOR

En date du :

Projet M1603	FICHE PROJET	Appel à projet 2015-2016

Dissection virtuelle à partir d`acquisitions de scanner surfacique: un nouvel outil d'enseignement de l`Anatomie et d`entrainement chirurgical

I. Descriptif du projet :

La dissection cadavérique conventionnelle est un outil essentiel pour l`apprentissage de l'Anatomie Chirurgicale, et devrait être un préalable obligatoire à la réalisation de tout geste sur le vivant. Toutefois, le nombre croissant d`étudiants en médecine et le nombre limité de cadavres disponibles ont conduit à une réduction de la pratique de la dissection, appelant au développement de nouveaux outils d'enseignement. Plusieurs publications récentes ont proposé d`utiliser des reconstructions numériques tridimensionnelles (3D) à partir d'acquisitions de scanner surfacique. Les acquisitions sont généralement effectuées à la fin de la dissection, ce qui ne permet pas à l`étudiant d'analyser les différentes étapes de dissection ou de comprendre les rapports en profondeur entre toutes les structures anatomiques. Notre projet est de développer un outil de dissection virtuelle dynamique applicable à tous les organes non cavitaires (membres, cou, face).

Le principe est de superposer des acquisitions de scanner surfacique réalisées sur différents niveaux de dissection pour créer un environnement quadridimensionnel (3D + temps) précis et réaliste. Cet environnement pourrait être implémenté d'outils de dissection, avec le développement d'un logiciel dédié. Il offrirait la possibilité de répéter et d`enregistrer les gestes réalisés et d`évaluer la courbe d`apprentissage.

La dissection virtuelle 4D peut être utilisée comme un véritable outil de formation interactif pour les étudiants en Médecine, et plus particulièrement des internes de chirurgie. Si elle n'est pas adaptée à l'étude des organes intra thoraciques, intra abdominaux ou intracrâniens in situ (zones d'ombres non accessibles aux faisceaux lumineux du scanner), elle est en revanche applicable à tous les membres, au cou, et à l'anatomie de surface du tronc et du crâne. Un système d'évaluation de l'interne en 3 étapes basé sur cet outil pourrait être envisagé, chaque étape conditionnant l'accès à la suivante : 1- validation des connaissances en anatomie chirurgicale sur le module de dissection virtuelle, 2- formation chirurgicale sur cadavre, 3- mise en application séniorisée chez le vivant, respectant la recommandation éditée par la Haute autorité de Santé en 2012 : « jamais la première fois sur un patient ».

II. Réalisation :

Une étude préliminaire a été réalisée dans notre centre. Nous avons effectué une dissection cervicale bilatérale sur cadavre frais, avec identification de 8 plans anatomiques d'intérêt pour la numérisation. Chaque plan a été numérisé à l`aide du scanner laser 3D de surface (Artec Spider ™) prêté pour l'occasion. Ce dispositif a une précision géométrique allant jusqu`à 0,05 mm, et permet également l'acquisition de couleurs en haute définition. Comme il ne nécessite pas de placer des marqueurs, le processus de numérisation est rapide et a pu être facilement intégré à un protocole de dissection classique. Après traitement numérique des acquisitions par un ingénieur, les 8 niveaux obtenus ont pu être superposés grâce au logiciel Meshlab®.

Nous souhaitons dans un premier temps appliquer ce protocole à différentes régions anatomiques, et dans un deuxième temps exploiter les acquisitions obtenues avec la participation d`un stagiaire informatique pour développer l`outil pédagogique.

Nous sollicitons donc l'aide de l'UNF3S pour:

-l'achat d'un scanner laser 3D de surface: 2600 euros (sur les 15700 que coute l`Artec Spider™)

-l`achat d`une carte graphique: 1000 euros

-l`acquisitions et le traitement des images par un ingénieur: 4600 euros

-le développement d' une interface numérique par un stagiaire informatique: 1800 euros