Rapport

Tag Thunger

le 1er mars 2024

Auteurs
Romain Cailly,
Mohamed Chergui,
David Guo,
Thomas Seng,
Mohamed-El-Mokhtar Sidi-Abdallah

Remerciements

Nous souhaitons exprimer notre profonde gratitude envers nos encadrants, Fabrice Maurel et Gaël Dias, pour leur soutien continu et leur implication tout au long de ce projet de Recherche en Immersion. Nous les remercions également pour leur compréhension et leur bienveillance, nous permettant de réaliser ce travail sous la forme d'un projet intensif de deux semaines, malgré toutes les contraintes que cette décision leur a imposées.

Nous remercions également les doctorants et chercheurs François Ledoyen et Nilesh Tete, pour leurs accompagnements et leurs conseils, leurs explications sur le côté technique de l'outil déjà existant et les besoins des non-voyants pour fournir un logiciel efficace.

Table des matières

État de l’art

1. Contexte

La navigation sur Internet est très largement visuelle. Seules 10 % des pages web sont accessibles aux personnes malvoyantes. C’est dans ce contexte que les chercheurs et ingénieurs du GREYC ont consacré leurs efforts à plusieurs concepts innovants pour améliorer l’accessibilité au web des personnes malvoyantes.

2. Solutions existantes

Au cours des dernières années, le laboratoire GREYC a travaillé sur deux approches distinctes, menant à des innovations prometteuses : ***TactiNET*** et ***Tag Thunder***.

a. TactiNET

Dans le domaine de l’accessibilité numérique, une première approche innovante émerge, mettant à profit l’utilisation de dispositifs tactiles. Cette méthode se base sur la détection des niveaux de lumière générés par la disposition des zones de texte au sein d’un document. L’ingéniosité de cette approche réside dans la conversion astucieuse de ces variations de luminosité en vibrations perceptibles par le lecteur non-voyant. Ces vibrations offrent une guidance précise, permettant ainsi à l’utilisateur de s’orienter de manière intuitive vers la zone souhaitée.

Cette approche vise à transcender les limites de la navigation traditionnelle en proposant une expérience immersive et inclusive. Elle aspire à être intégrée de manière synergique avec d’autres modalités sensorielles, telles que l’audition et le toucher, pour enrichir l’expérience utilisateur. Cette symbiose entre différentes modalités sensorielles ambitionne la création d’une navigation web plus inclusive, où les personnes non-voyantes peuvent explorer et interagir de manière plus naturelle avec le contenu en ligne, dépassant ainsi les barrières qui entravent actuellement leur expérience sur la toile.

b. Tag Thunder

i. Description

Pour résoudre efficacement le problème d’accessibilité aux sites web pour les malvoyants, une seconde approche novatrice consiste à segmenter les différentes zones de la page et à assigner une série de mots-clés distincts à chacune de ces zones. Ensuite, grâce à une fonction de vocalisation, l’utilisateur peut entendre une représentation audio spatialisée en 3D en fonction de la position du curseur.

Cette approche offre une expérience immersive unique, où les mots-clés sont perçus dans l’espace autour de l’utilisateur, avec une spatialisation sonore précise. Ainsi, en se déplaçant virtuellement dans cet environnement sonore, l’utilisateur peut identifier et sélectionner les mots-clés qui l’intéressent le plus. Cette interaction, combinant l’audio spatialisée et la navigation intuitive, vise à rendre la découverte du contenu web plus engageant et accessible pour les personnes non-voyantes.

ii. Prémices

Une première version du projet, proposé par Nilesh Tete, permettait d'avoir une localisation basique sur un exemple de page web simple. Le volume sonore était toutefois peu variable, et la piste audio n'était pas toujours jouée à la bonne vitesse. De plus, il y avait une latence trop élevée pour pouvoir exploiter la fonctionnalité. Elle permettait également de lire une piste audio selon la présence du curseur dans une certaine zone. Il s'agissait d'une version utilisable sur une page statique prédéfinie, et son code n'était pas exploitable.

3. Contribution du projet

Dans le cadre de notre projet, notre objectif est de présenter une première implémentation d'un plugin offrant un accès à toute page web aux personnes souffrant d'un handicap visuel. Ce plugin, une fois activé, a la capacité de vocaliser les thèmes présents dans les différentes zones de texte d'un site web.

Cette avancée représente une contribution significative pour surmonter les défis auxquels sont confrontés les utilisateurs, non-voyants lors de leur navigation sur Internet. En intégrant cette fonctionnalité, notre ambition est de rendre l'expérience web plus inclusif, en fournissant une compréhension verbale des contenus visuels cruciaux, facilitant ainsi l'accès à l'information pour tous les utilisateurs, indépendamment de leur capacité visuelle.

Solutions apportées

Notre solution repose sur une extension Mozilla. L'extension récupère l'URL de la page et l'envoie à l'API développée par François Ledoyen puis retravaillée par notre équipe. Elle reçoit alors, en sortie du pipeline, un fichier JSON permettant d'effectuer les traitements à réaliser pour proposer la fonctionnalité du Tag Thunder.

1. Répartition des tâches

Notre travail s'est décomposé en plusieurs parties.

La première partie est le travail de modification et de correction effectué sur le pipeline créé par François Ledoyen. Ce travail a été mené par Mohamed Chergui, David Guo et Mohamed-El-Mokhtar Sidi-Abdallah.

La seconde partie concerne la création de l'extension Mozilla permettant d'utiliser l'outil Tag Thunder par l'utilisateur aisément, réalisé par Romain Cailly, David Guo et Thomas Seng.

La troisième partie, représentée par l'implémentation des différentes fonctionnalités du Tag Thunder, se subdivise en plusieurs catégories. La réalisation de la visualisation de la segmentation a été faite par Romain Cailly et Thomas Seng, la spatialisation de l'audio par Romain Cailly, et la synthèse de la voix par Mohamed Chergui et Mohamed-El-Mokhtar Sidi-Abdallah. Romain Cailly s'est ensuite chargé de la fusion des modules.

2. Description fonctionnelle

a. Visualisation de la segmentation

Pour visualiser la segmentation, nous utilisons la donnée xpath fournie par le fichier JSON pour récupérer la balise correspondant à la zone déterminée par l'algorithme de segmentation.

Nous ajoutons du style CSS au pseudo-élément :before de la zone afin de rajouter une couleur et le mot-clé extrait de la segmentation sans altérer la structure de l'arbre DOM de la page (et donc rendre potentiellement obsolète le xpath). La couleur est définie en fonction du thème : si deux zones ont le même thème, alors elles partageront la même couleur.

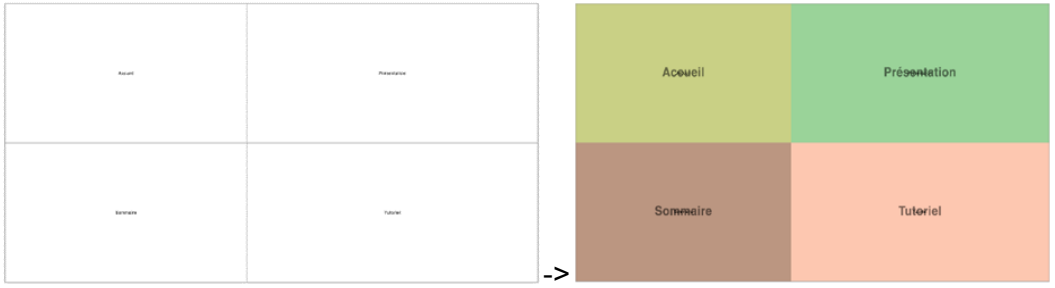

Figure 1 : Segmentation visuelle implémentée par l'extension

b. Spatialisation de l'audio

i. Principe

Après avoir activé l'option de Spatialisation, l'utilisateur doit cliquer sur la page afin de déclencher la lecture des pistes audio (cela est dû à une sécurité de Firefox)

Les distances entre les centres de chaque zone et le curseur sont calculés automatiquement. Selon la proximité entre le curseur et une zone, le thème associé aura un volume sonore variable.

ii. Génération du son

La synthèse vocale se fait grâce à la librairie MeSpeak qui permet de créer des HTMLAudioElement à partir d'un texte.

Documentations utiles :

- https://www.masswerk.at/mespeak/#:~:text=js
- https://github.com/btopro/mespeak/tree/master

iii. Spatialisation du son

Le son est spécialisé grâce au Web Audio API. Cet API, implémentée de base dans Firefox permet d'agir de différentes façons sur les sons émis par la page et de notamment spécialiser un son. Pour cela, il faut créer un contexte audio et l'utiliser pour créer des sources audios et définir un écouteur. L'écouteur correspond à la souris. Les sources audios correspondent quant à elles aux différentes zones de la segmentation.

3. Mise en place de la solution

a. Mise en place de l'extension Mozilla

Prérequis :

- Mozilla Firefox

Ajout de l'extension au navigateur :

- Ouvrir Mozilla Firefox ;
- Taper dans l'URL: about:debugging ;
- Aller dans la section *Ce Firefox* ;
- Cliquer sur Charger en module complémentaire temporaire… ;
- Sélectionner le fichier manifest.json de l'extension.

b. Installation

Prérequis :

- Python (version 3.9 ou 3.10) ;
- Poetry ;
- Make ;
- Screen ;
- Docker/Docker desktop.

Étape d'installation :

1. Cloner l'un des projets suivants :
    a. Git du projet de Francois Ledoyen - https://git.unicaen.fr/imalang/tagthunder ;
    b. Git de notre projet - https://git.unicaen.fr/guo221/tagthunder.
2. Taper la commande : cd tagthunder ;
3. Taper la commande : make install ;
4. Ouvrir deux terminaux séparément :
    a. Taper la commande : make run-api ;
    b. Taper la commande : make run-crawler.

Il existe une règle dans le Makefile (à la racine du projet) permettant d'exécuter le crawler en local, ce qui est utile pour faire des tests sur un serveur local (exemple : page web sur un serveur local). Au lieu d'entrer la commande make run-crawler, il faut utiliser la commande : make run-crawler-local.

c. Utilisation du *pipeline*

Localement, pour accéder à l'UI *Swagger* pour API *Tag Thunder*, il faut au préalable installer le projet *Tag Thunder* puis aller à l'adresse suivante http://127.0.0.1:8000/.

Le *pipeline* est composé de plusieurs modules accessibles au moyen d'une API. Ci-dessous, sont listés les différents endpoints disponibles, ainsi que leur entrée et leur sortie.

Module	**Route**	**Type de requête**	**Entrée**	**Sortie**
Augmentation	/ask/augmentation	POST	URL d'un site web	HTMLP
Cleaning	/ask/cleaning	POST	HTMLP	HTMLPP
Segmentation	/ask/segmentation	POST	HTMLPP	JSON
Extraction	/ask/extraction	POST	JSON	JSON
Pipeline	/ask/pipeline	POST	URL d'un site web	JSON

- ***Augmentation*** : Récupération des informations sur les éléments de la page ;

- ***Cleaning*** : Extraction et préparation des informations visibles de la page ;
- ***Segmentation*** : Découpage de la page en plusieurs zones ;
- ***Extraction*** : Détermination des thèmes de chaque zone à partir d'une page segmentée.

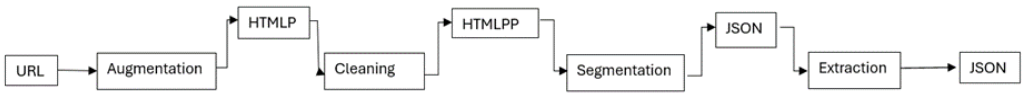

Figure 2 : Du module de l'augmentation jusqu'au module de l'extraction

Le module ***Pipeline*** permet d'appeler les modules précédents de manière séquentielle.

4. Fonctionnement de l'extension

Dans cette partie, nous allons décrire le fonctionnement interne de l'extension et sa structure.

a. Décomposition

Notre extension contient une fenêtre pop-up. Cette fenêtre a son propre code HTML et CSS décrivant son interface ainsi que son code JavaScript décrivant sa logique.

Notre extension doit être capable d'ajouter des sons à la page courante et du style CSS pour visualiser la segmentation. Or, le code JavaScript de la fenêtre pop-up ne peut interagir avec la page courante. Il est donc nécessaire d'injecter du code JavaScript dans la page pour que celui-ci puisse interagir avec la page.

Notre extension se décompose donc en deux parties :

- **Extension pop-up :** injecte le code JavaScript dans la page courante et contrôle les actions de ce code (activer/désactiver le son spatialisé et/ou la visualisation de la segmentation).
- **Extension injectée :** implémente la logique d'implémentation du son spatialisé et de la segmentation visuelle.

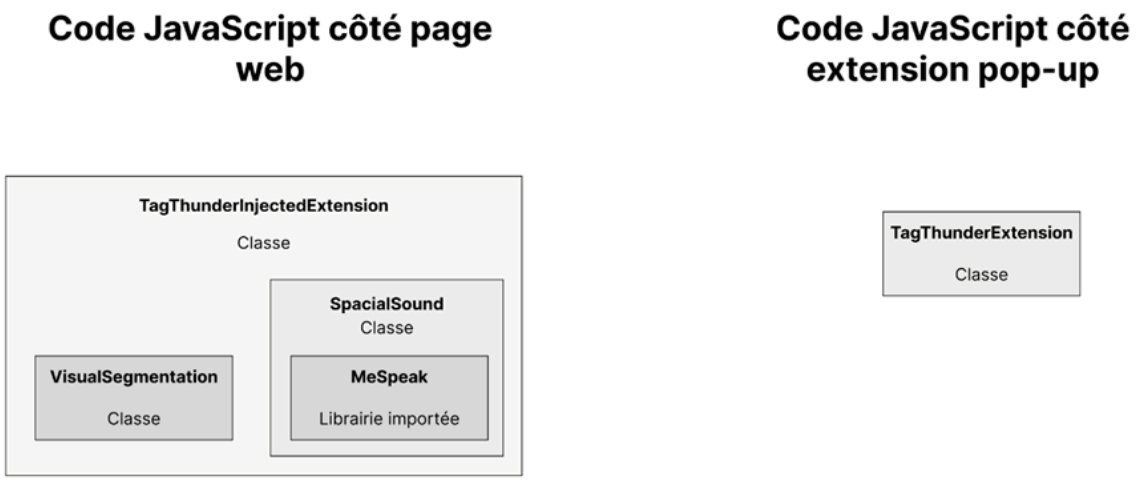

Figure 3 : Décomposition de l'extension

b. Contrôle de l'extraction

C'est l'extension pop-up qui contrôle l'extension injectée en lui indiquant les différents changements d'état (activer/désactiver le son spatialisé et/ou la visualisation de la segmentation) induit par les cliques de l'utilisateur sur les différents boutons de la fenêtre pop-up.

Pour cela, on utilise l'API de message implémentée par défaut dans les extensions Firefox et qui permet d'envoyer des messages à une page spécifique du navigateur. Ainsi, lorsque l'utilisateur souhaite par exemple activer le son spécialisé, alors un message "**enable-spacialsound**" est envoyé à l'extension injectée dans la page courante qui réagit à ce message en activant le son spécialisé.

c. Persistance de l'état

Lorsque l'on quitte la fenêtre pop-up, celle-ci est arrêtée complètement. Il est donc nécessaire de mettre en place un système de persistance de l'état de l'extension sur une page donnée.

Pour cela, nous avons défini des attributs sur la balise body de la page permettant de sauvegarder l'état actuel de l'extension sur cette dite page (cf capture d'écran ci-dessous).

Au nombre de quatre, ils permettent respectivement :

- **tagthunder** : l'extension a été injectée dans la page,
- **tagthunder-running** : l'extension est activée,
- **tagthunder-visual-segmentation** : la segmentation visuelle est activée,
- **tagthunder-spacial-sound** : le son spécialisé est activé.

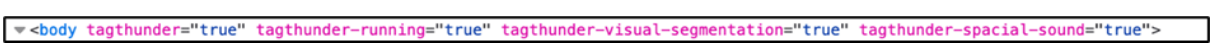

Figure 4 : élément body de la page web

Ainsi, lorsque l'on ouvre la fenêtre pop-up de l'extension, celle-ci va récupérer les différents attributs présents sur la balise body de la page pour connaître l'état de l'extension actuelle et ne pas injecter par exemple deux fois le code dans une même page.

5. Contraintes et problèmes rencontrés

a. Absence des champs dans le JSON de sortie

Lors de l'utilisation du pipeline, nous nous sommes rendu compte qu'après le refactor de François le Doyen, plusieurs champs ne sont plus présents, car le pipeline a été adapté pour l'entreprise *Accessman*.

Ces champs sont :

- Les coordonnées des différentes boxes
- Le champ *datacleaned*

Ces champs sont cependant disponibles dans l'HTLMPP, mais leur récupération est compliquée et nécessite des regex

Ajout du *XPath*

Lors de l'étape de Segmentation, un fichier JSON est généré, composé de plusieurs zones. Chaque zone peut comporter un ou plusieurs XPath. L'objectif de cette étape était d'ajouter un champ XPath contenant la liste des XPath de chaque zone, simplifiant ainsi le travail de spatialisation du son en fournissant directement un champ XPath dédié dans le JSON.

Pour réaliser cela, nous avons introduit un attribut *XPath* de type liste de chaînes de caractères dans la classe *schema.Zone*, ainsi qu'une méthode ***extract_xpaths_from_htmlpp()*** dans la classe *responses*. Cette dernière correspond à la sortie de l'étape de segmentation. La méthode permet de récupérer la liste des *XPaths* contenus dans le *htmlpp* en utilisant un filtre avec une expression régulière. Cette méthode est appelée dans la méthode Zone de cette même classe afin de créer une instance de la classe *schema.Zone* avec les paramètres appropriés, notamment les *XPaths*.

b. Erreurs communes lors de l'installation du projet TagThunder

i. Fichier html-augmentation-1.0.0.tgz introuvable par npm

Piste de solution 1

- Désinstaller et réinstaller : *node*, *npm*, *python*.

Piste de solution 2

- Taper la commande *make install* dans le répertoire **taghunder** afin d'installer tous les paquets nécessaires (autre que le dossier compressé **html-augmentation-1.0.0.tgz**) ;
- Taper la commande *npm pack* dans le dossier **html-augmentation** afin de générer le **html-augmentation-1.0.0.tgz** ;
- Taper la commande *npm install html-augmentation-1.0.0.tgz*.

Piste de solution 3

Avant d'entrer la commande *make install* à la racine du projet :

- Aller dans le répertoire **tagthunder/tagthunder/javascript/puppeteer-crawler** ;
- Taper la commande *npm install*.

Cela permet de générer le fichier **html-augmentation-1.0.0.tgz** en avance avant de construire tout le projet.

ii. Erreur lors du build

Cette erreur peut avoir lieu durant l'étape build du projet sous Windows. Elle est probablement liée au fait que docker crée d'abord tout à l'extérieur et fait ensuite un COPY vers l'intérieur.

Voici à quoi pourrait ressembler l'erreur :
"ERROR: error during connect: this error may indicate that the docker daemon is not running: Get "http://%2F%2F.%2Fpipe%2Fdocker_engine/_ping": open //./pipe/docker_engine. Le fichier spécifié est introuvable."

Solution :

Une solution potentielle pourrait être de tout créer dans le *docker* pour ne pas avoir besoin de tout copier par la suite.

c. Découverte de la structure d'une extension Mozilla

N'ayant aucune connaissance sur les extensions Mozilla, la compréhension de sa structure nous a pris beaucoup de temps. Toutefois, cette compréhension était nécessaire afin de pouvoir concevoir au mieux celle-ci, et pouvoir implémenter les diverses fonctionnalités attendues.

d. Limite inhérente au navigateur Firefox

Une sécurité du navigateur Firefox empêche la lecture d'un fichier audio si aucune interaction de l'utilisateur avec la page n'a été effectuée au préalable. Ainsi, l'audio spatialisé généré par notre application, malgré son activation, ne se joue que si l'utilisateur a cliqué au moins une fois sur la page.

Poursuite du projet

1. DevOps

a. Système d'exploitation

Des problèmes d'installation et de compilation du logiciel peuvent survenir, dus à l'utilisation de NodeJS.

Notre équipe a testé d'installer l'application sur différentes machines et différents systèmes d'exploitation (PC personnel) :

- MacOS
- Windows 11
- Linux (Ubuntu)

Nous n'avons réussi à faire fonctionner l'application que sous Ubuntu. Si cela ne marche pas, désinstallez les prérequis et réinstallez les prérequis (voir section d'avant)

b. Conteneurisation de l'application

Dans la section Installation, de nombreuses lignes de commande sont à entrer dans le terminal pour que l'utilisateur installe le logiciel (en supposant les prérequis validés), ce qui peut être un frein à l'utilisation du logiciel.

Il faudrait qu'une conteneurisation soit mise en place, par exemple avec Docker et Ansible, pour que l'application soit déployée en une seule ligne de commande sans prérequis. Il faut voir cela avec la DSI, pour savoir quel outil d'orchestration ou de conteneurisation est recommandé.

c. Vulnérabilités

Lors de l'installation de l'application sur l'ordinateur, plusieurs vulnérabilités de l'application sont détectées :

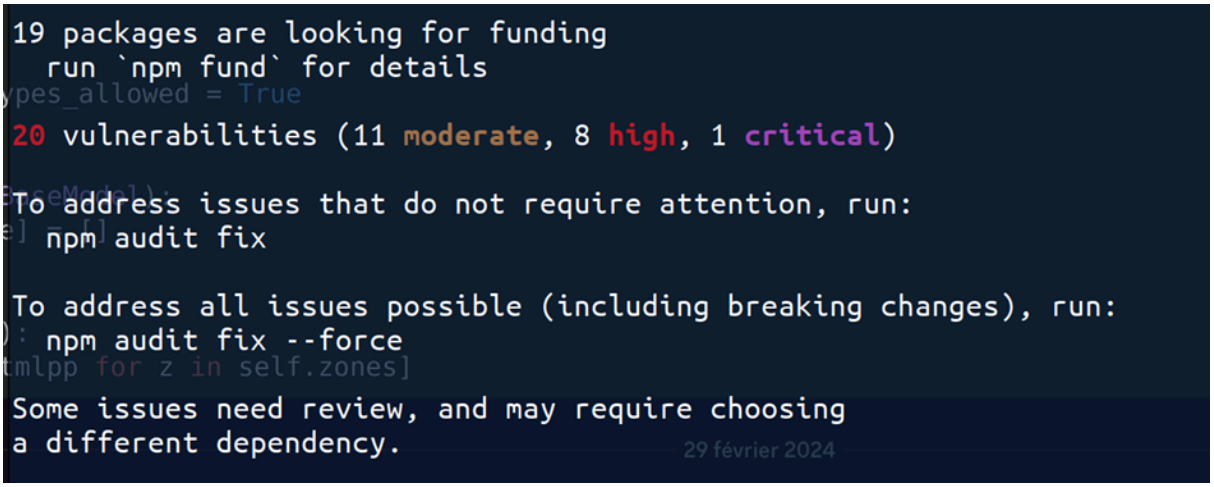

Figure 5 : Vulnérabilités lors de l'utilisation de npm

Il faudrait régler ces problèmes de vulnérabilité.

2. Pipeline

Pendant l'étape de nettoyage, pour alléger le fichier de sortie du module, certaines données du fichier HTML sont supprimées pour des raisons d'optimisation. Cependant, lors de notre discussion avec M. Fabrice Maurel, il a exprimé le souhait d'avoir un pipeline qui saute cette étape. En effet, des données supprimées lors de cette étape pourraient être pertinentes pour la suite du processus.

Nous avons tenté de modifier le code pour contourner cette étape. Cependant, il semble que les étapes du pipeline soient trop étroitement liées et que le code manque de clarté, notamment en ce qui concerne le rôle de chaque fichier, en l'absence de commentaires explicatifs. Par manque de temps, nous n'avons pas pu accomplir cette tâche. Néanmoins, nous sommes convaincus que la personne qui travaillera sur ce problème par la suite pourra y parvenir facilement avec des indications de François Ledoyen sur la manière dont les étapes du pipeline sont connectées entre elles.

Par la suite, nous allons voir quelques idées et pistes afin de pouvoir mettre en place un pipeline adaptable selon le choix du développeur. C'est-à-dire passer par l'étape de cleaning ou non.

a. Ajout d'un champ *datacleaned*

Nous avons pensé à utiliser un booléen *datacleaned* lors de la création du pipeline afin de préciser si l'étape cleaning a été réalisée ou pas. L'avantage de cette approche, c'est qu'elle permet de garder les deux approches, et de basculer de l'une à l'autre facilement.

b. Paramétrage de la pipeline

Si le choix d'enlever l'étape cleaning n'est pas définitif, alors il serait plus judicieux de faire en sorte que le pipeline soit paramétrable, dans le sens où on pourrait faire tourner le pipeline avec certains modules, ou algorithmes de notre choix.

c. Adaptation du pipeline pour la recherche

Si le choix d'enlever l'étape cleaning est définitif, nous pourrons opter pour une approche plus radicale qui consisterait à virer tout le module cleaning. Ainsi, la sortie du module augmentation serait connectée directement à l'entrée du module segmentation.

3. Amélioration de l'extension

a. Implémentation d'options pour l'audio spatialisé

La fonctionnalité de l'audio spatialisé doit pouvoir être davantage configurable. Voici une liste des paramètres qu'il doit être possible de configurer :

- Vitesse de lecture des thèmes ;
- Fréquence de répétition des thèmes ;
- Fonction d'atténuation du volume sonore en fonction de la distance entre le curseur et la zone.

b. Amélioration de la gestion des zones

Notre solution se base sur l'hypothèse que chaque zone correspond à une balise HTML unique, ce qui est une simplification de la réalité. En effet, une zone peut correspondre à un groupement de plusieurs balises. Un travail doit donc être fait afin de corriger cette approximation.

c. Type de spatialisation de l'audio

Concernant la spatialisation de l'audio, notre implémentation s'appuie sur de l'audio binaural, et non de l'audio 3D. Il est donc possible de distinguer les sons provenant de la gauche de ceux provenant de la droite, mais impossible d'avoir une information par rapport à la hauteur et la profondeur de la source.

Conclusion

Le projet nous a permis de découvrir beaucoup d'éléments intrinsèques au travail de recherche. L'approche d'un chercheur est très différente de celle d'un ingénieur, et la conception d'un projet en est largement impactée. Quand l'ingénierie se focalise sur une solution optimisée, la recherche se focalise plus sur une solution fonctionnelle et adaptée aux utilisateurs, avec des retours utilisateurs fréquents.

Nous avons également pu perfectionner nos connaissances techniques, notamment avec la découverte de la conception et de l'utilisation d'une extension Mozilla. Sa structure étant particulière, nous avons pu avoir une première approche de ce type de technologies. Le travail de correction du pipeline a aussi été bénéfique dans notre apprentissage. L'analyse efficace de code préalablement implémenté dans le cadre de la poursuite d'un projet est une qualité indispensable à un ingénieur.

Si nous avons des regrets concernant l'échec de la liaison entre l'extension et le pipeline, nous sommes globalement satisfaits de ce que nous avons pu produire en deux semaines. Le travail mené sur l'extension est qualitatif, nous avons soulevé des points critiques dans la conception du pipeline, et nous avons apporté certaines corrections en local, qu'il sera assez facile à déployer.

Les retours de nos encadrants, Fabrice Maurel, Gaël Dias et Nilesh Tete, ont été assez positifs. Même si les livrables doivent être retravaillés, certaines fonctionnalités sont imparfaites et d'autres manquantes, le travail effectué est plutôt satisfaisant pour une première version produite en temps limité et destinée à être reprise pour la poursuite du projet.