



Stage M2

Etat de l'art logiciel traitement vidéo

Benjamin Serva
Master 2 IMAGINE
Université de Montpellier

Encadrants :

Olivier Strauss & William Puech & Frédéric Comby

Contents

1	Introduction	3
2	Fonctionnalités souhaité	3
3	Different Logiciel pour de la lecture/traitement de vidéo	3
3.1	VLC	3
3.2	QuickTime	4
3.3	Screen	5
3.4	IINA	6
3.5	MPV	7
3.6	DaVinci Resolve	7
3.7	Comparaison	8
4	Article	8
4.1	KinoAI	8
5	Bibliothèques INA	9
5.1	inaSpeechSegmenter	9
5.2	inaFaceAnalyser	10
5.3	speaking-clock-detection	10
6	Méthodes	10
6.1	Transcription d'un audio en texte	10
6.1.1	Alternative 1 : SpeechRecognition	11
6.1.2	Alternative 2 : Google Cloud Speech-to-Text	11
6.2	Gestion et extraction d'informations intégrées dans les fichiers vidéo	11
6.3	Reconnaissance de texte dans les vidéos : Exploitation d'OCR (Reconnaissance Optique de Caractères)	12
6.4	Segmentations de scènes	13
6.4.1	Shot Boundary Detection (SBD)	13
6.5	PySceneDetect	15
6.6	YOLO : You Only Look Once	16
6.6.1	Détection d'objets et identification	16
6.6.2	Poursuite de cible	17
6.7	Détection de Visage	17
6.8	Optimisation des Performances	19
7	Choix des langages de programmation	19
7.1	Python	19
7.2	Qt	20
8	Définition	20
8.1	Étude iconographique	20
8.2	Profilmique	20
8.3	Interopérabilité	20
9	Références	21

1 Introduction

Cette étude de l'art a pour objectif, dans un premier temps, d'exposer les besoins liés au logiciel, puis, dans un second temps, de référencer les principaux logiciels de traitement vidéo existants ainsi que les différentes techniques essentielles au traitement des vidéos.

2 Fonctionnalités souhaité

Volet 1

- interface ergonomique et intuitive inspirée de Screen
- lecture de plusieurs vidéos en simultanément et synchroniquement
- zoom
- affichage d'image fixes, texte
- lecture de son
- système de calque/grille
- outil de segmentation
- interface d'annotation manuelle
- capture images fixe et capture sonore
- découpage et encodage de segment source vidéo en format commun
- recherche **iconographique**^{*} en ligne

Volet 2

Premier temps

- détection intelligente des plans.
- types de transitions.
- échelles de cadrage.
- types de mouvements optiques.
- paramètres colorimétriques.

Second temps Reconnaissance automatisée de contenus "**profilmiques**"^{*} spécifiques (types de situation, comportements d'objets ou d'acteurs, d'objets sonores).

3 Différent Logiciel pour de la lecture/traitement de vidéo

3.1 VLC

PLateforme Multi plateforme et disponible en 69 langues.

Détail VLC est un lecteur multimédia libre et open source, reconnu pour sa compatibilité étendue avec la plupart des formats audio et vidéo sans nécessiter de codecs supplémentaires. Il permet également la conversion de fichiers multimédias et l'extraction de la piste audio d'une vidéo.

Il prend en charge la gestion avancée des sous-titres, incluant le chargement automatique, la synchronisation et la personnalisation de l'affichage.

L'un des points forts de VLC est sa robustesse : il peut lire des fichiers multimédias incomplets, endommagés ou en cours de téléchargement. Il supporte également la lecture en streaming à partir de flux réseau, y compris les protocoles HTTP, RTP, RTSP et HLS.

VLC offre une large gamme de filtres et d'effets vidéo, tels que la distorsion, la rotation, l'inversion, le désentrelacement, l'ajustement des couleurs, l agrandissement et le redimensionnement. Il propose aussi des effets audio, comme l'égalisation et la spatialisation du son.

Enfin, VLC dispose d'une API qui permet de tirer parti de ses fonctionnalités, notamment pour l'intégration dans des solutions de lecture ou de traitement vidéo.

Language de programmation utilisé pour la conception

- Language : C, C++ et objective-C.
- Interface graphique : Qt, ncurses et Cocoa (pour développer sur Apple).

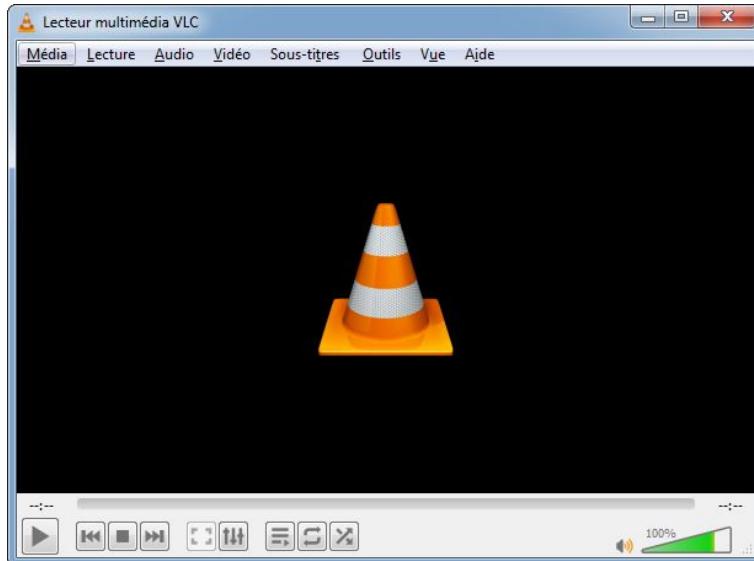


Figure 1: Interface de l'application VLC

3.2 QuickTime

Plateforme Disponible uniquement sur Windows et macOS.

Détail QuickTime est un lecteur multimédia développé par Apple, principalement utilisé pour la lecture de fichiers multimédias sur macOS et Windows. Il prend en charge une large gamme de formats audio et vidéo, notamment MOV, MP4, AAC et AIFF.

Il offre des fonctionnalités de lecture avancées, notamment la prise en charge des chapitres, des sous-titres et du réglage de la vitesse de lecture.

QuickTime est également connu pour ses capacités d'édition de base, permettant de découper, fusionner et convertir des vidéos.

Il intègre des codecs propriétaires optimisés pour l'écosystème Apple et offre une excellente qualité de lecture des fichiers multimédias.

L'API QuickTime permet aux développeurs d'intégrer ses fonctionnalités dans des applications tierces.

Langage de programmation utilisé pour la conception

- Langage : C.
- Interface graphique : Cocoa (macOS), Windows API (Windows).



Figure 2: Interface de l'application QuickTime

3.3 Screen

Payant → 176€.

Plateforme Disponible sur macOS.

Détail Screen est une application de traitement et de gestion vidéo développée par VideoVillage, spécialement conçue pour les professionnels du cinéma et de l'audiovisuel. Elle permet de visualiser, analyser et annoter des vidéos avec une grande précision.

L'une de ses fonctionnalités principales est la gestion de plusieurs pistes vidéo simultanées, ce qui permet une comparaison et une analyse côte-à-côte. Screen prend également en charge le **zoom temporel et spatial** pour examiner des détails fins dans les vidéos.

Il permet d'afficher et d'annoter des images fixes, de rajouter des textes, des repères et d'effectuer des découpages dans les vidéos. Son interface ergonomique facilite l'ajout d'annotations manuelles, et les outils de **calque et de grille** permettent un alignement précis des éléments.

L'application supporte aussi le **traitement audio** et la gestion des sous-titres, avec des outils de synchronisation et de personnalisation des fichiers.

Screen dispose d'une **API flexible** permettant une intégration dans des workflows de production vidéo complexes, ainsi que des options avancées de **segmentation et de codage de segments vidéo** en différents formats standards.

Langage de programmation utilisé pour la conception

- Langage : C++, Python.
- Interface graphique : Qt (multi-plateforme), Cocoa (macOS).



Figure 3: Interface de l'application Screen

3.4 IINA

Plateforme Disponible sur macOS et basé sur le lecteur open-source mpv.

Détail IINA est un lecteur multimédia moderne pour macOS, conçu pour offrir une expérience fluide et intuitive. Il prend en charge une large gamme de formats audio et vidéo, ainsi que le streaming. Parmi ses fonctionnalités, on trouve un mode Picture-in-Picture, l'intégration avec la Touch Bar, ainsi qu'un mode sombre natif. L'application offre également une interface utilisateur moderne et personnalisable, avec une prise en charge complète des raccourcis clavier.

Langage de programmation utilisé pour la conception

- Langage : Swift.
- Interface graphique : Cocoa (macOS).

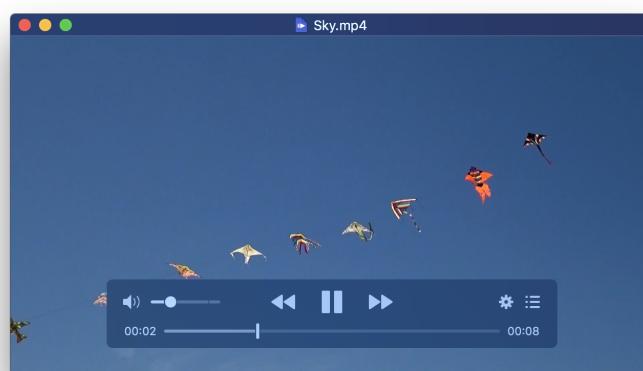


Figure 4: Interface de l'application Iina

3.5 MPV

Plateforme Disponible sur Windows, macOS et Linux.

Détail MPV est un lecteur multimédia open-source dérivé de MPlayer et mplayer2, offrant une interface minimalistre et une grande flexibilité. Il prend en charge une large variété de formats audio et vidéo, notamment MKV, MP4, AVI, AAC et FLAC. Il est particulièrement apprécié pour ses performances optimisées et son intégration avec des scripts Lua et Python, permettant une personnalisation avancée. MPV offre un support avancé des sous-titres, du rendu HDR et de l'accélération matérielle via Vulkan, OpenGL et Direct3D. Sa conception modulaire permet aux développeurs de l'intégrer facilement dans des applications tierces via son API libmpv.

Langage de programmation utilisé pour la conception

- Langage : C.
- Interface graphique : Pas d'interface native, contrôlé via CLI ou interfaces tierces.

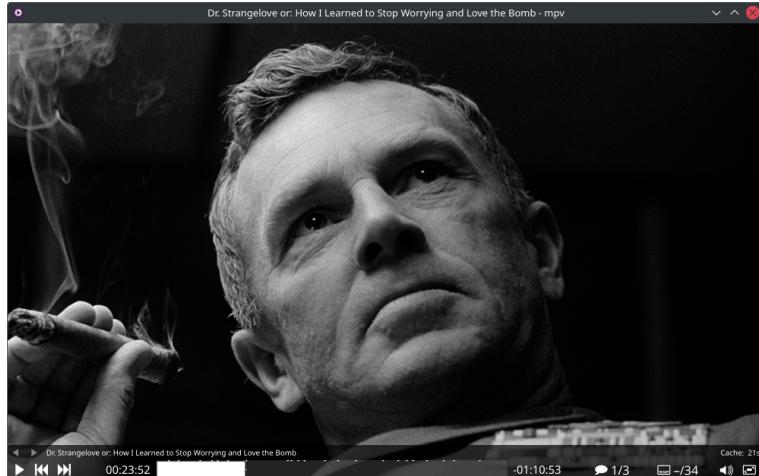


Figure 5: Interface de l'application MPV

3.6 DaVinci Resolve

Version Studio Payante → 354€.

Plateforme Disponible sur Windows, macOS et Linux.

Détail DaVinci Resolve est un logiciel professionnel de montage, d'étalement et de post-production vidéo développé par Blackmagic Design. Il est largement utilisé dans l'industrie cinématographique et audiovisuelle pour sa précision colorimétrique et ses outils avancés de correction des couleurs.

Il prend en charge un large éventail de formats professionnels, notamment ProRes, DNxHD, RAW et H.264/H.265.

Outre le montage non linéaire, il intègre des fonctionnalités avancées comme le suivi de mouvement, la reconnaissance faciale et des outils d'intelligence artificielle grâce à DaVinci Neural Engine.

- Reconnaissance faciale : analyse de vidéo qui sort chaque personne présente dans celle-ci avec une image. Ensuite il suffit de nommer chaque personne et on peut avoir tous les extraits avec cette personne.
- Analyse des dialogues dans une vidéo (4min pour une vidéo de 1h), affichage de tous le texte est possible d'accéder à la séquence vidéo qui contient une phrase.
- Capable de trier les sons, vidéos etc ... en catégorie.
- Détection de scène automatique

La version Studio offre des capacités supplémentaires comme l'exportation en 8K, l'optimisation du rendu GPU et la collaboration multi-utilisateurs.

Langage de programmation utilisé pour la conception

- Langages : C++, Python (pour les scripts et plugins).
- Interface graphique : Qt.



Figure 6: Interface de l'application DaVinci Resolve

3.7 Comparaison

Logiciel	Plateforme	Langages utilisés	API	IA	Payant
VLC	Multi-plateforme	C, C++, Objective-C	Oui	Non	Non
QuickTime	macOS, Windows	C	Oui	Non	Non
Screen	macOS	C++, Python	Oui	Non	Oui
IINA	macOS	Swift	Non	Non	Non
MPV	Multi-plateforme	C	Oui	Non	Non
DaVinci Resolve	Multi-plateforme	C++, Python	Oui	Oui	Oui/Non

Table 1: Comparaison des logiciels de traitement vidéo

4 Article

4.1 KinoAI

Contexte C'est un logiciel novateur qui utilise l'IA pour analyser des captations vidéo fixes de répétitions.

Application Détection de personnages et calcul de cadrage automatique les plus pertinent.

Pour ce qui est de la détection de personnage la librairie OpenPose a été utilisé. Pour le découpage, des algorithme d'optimisation temporelle ont été utilisé.

Problématique : Pour 1h de captation on obtient 10h de découpage.

Donc il fallait limiter au maximum les prise de vue, l'application a été complètement retravaillé pour intégrer une version client-serveur avec un système de visualisation en ligne pour éviter une limite de temps sur les données fournies/traitées.

Système d'annotation avec deux modes:

- Mode Table : notation simplement importé et associé, la notation est mise en rouge quand l'action associé apparaît, notation cliquable pour visionner la scène associé.
- Mode Planche : choix des cadres en fonction des notations, séquence découpé en fonction des notations, plan large et plan choisi par l'ingénieur en fonction de la note.

Retour Prise en main compliqué et temps de traitement trop élevé.



Figure 7: Interface de l'application KinoAI

5 Bibliothèques INA

5.1 inaSpeechSegmenter

Language Elle est développée en python.

Pré-requis ffmpeg pour décoder n'importe quel type de format.

Fonctionnalités Proposées

1. Détection de genre du locuteur : homme ou femme (modèle de classification optimisé pour la langue française car entraîné dessus).

2. Capable de dire si un humain chante ou parle. Parole sur bruit ou parole sur musique sont étiquetées comme "Parole" et la voix chantée est étiquetée comme "Musique".

5.2 inaFaceAnalyser

Language Elle est développée en python.

Pré-requis cmake, FFmpeg et libgl1-mesa-glx.

Fonctionnalités Proposées Utilise un modèle de classification basé sur l'architecture ResNet50.

1. Prédiction de l'âge et du sexe à partir des visages détectés dans des images ou des vidéos.
2. Exportation des résultats sous différents formats, notamment des tableaux CSV, des flux vidéo annotés ou des sous-titres au format ASS.
3. Traitement optimisé pour l'analyse à grande échelle, permettant des performances élevées lors de campagnes de surveillance médiatique.
4. Réduction des biais liés au genre, à l'âge et à l'origine ethnique grâce à l'utilisation de modèles d'apprentissage approfondi entraînés sur des jeux de données diversifiés.

5.3 speaking-clock-detection

Language Elle est développée en python.

Pré-requis Ne fonctionne que sur des fichiers stéréo.

Fonctionnalités Proposées Cet outil peut être utilisé pour détecter sur quel canal une horloge parlante est présente. En sortie trois valeurs sont possibles :

- **SPEAKING_CLOCK_TRACK** : suivi par l'identifiant du canal (0 ou 1).
- **SPEAKING_CLOCK_NONE** : aucune horloge parlante détectée.
- **SPEAKING_CLOCK_MULTIPLE** : plusieurs horloge parlante détectée ce qui est considérée comme une erreur.

6 Méthodes

6.1 Transcription d'un audio en texte

Cette méthode sera utile dans le cas où aucune métadonnées (tel que des sous titres) ne sont associés à la vidéo.

6.1.1 Alternative 1 : SpeechRecognition

Moteur de reconnaissance vocale : Google Web Speech API.

Seulement le format **WAV** pris en charge. Il est capable de transcrire une multitude de langue.

Temps de d'exécution de 37 secondes pour une vidéo de 2 minutes et 7 secondes, avec une très bonne précision plus de 95% de bonne correspondance en sachant que le test a été effectué sur un audio d'une personne anglaise qui parle français.

Lien vers la vidéo dont j'ai extrais le contenu audio et convertis en .wav : Youtube - Le Parisien "le discours en français du Roi Charles III à Versailles"

Texte transcrit : *il nous incombe à tous de revigoriser notre amitié pour qu'elle soit à la hauteur des défis de ce 21e siècle monsieur le Président Madame Macron je ne serai pour dire à quel point mon épouse et moi-même sommes ravi d'être parmi vous ce soir au thème de la première journée de notre première visite d'État en France et combien nous sommes touchés par la magnifique accueil qui nous a été réservé une fois de plus la France et le peuple français nous ont témoigné un accueil chaleureux et une profonde gentillesse et nous leur ensemble très reconnaissant généralité nous rappelle que ma famille et moi-même avons été très ému par les hommages vendus en France à ma mère fait elle dans les funérailles ont eu lieu il y a une année je voudrais donc si vous ne le permettez porter un toast à monsieur le Président Madame Macron et le peuple français ainsi qu'à notre entente cordiale une entente pour le développement durable Christelle fidèle et constante à travers les siècles qui nous attendent contre vents et marée*

6.1.2 Alternative 2 : Google Cloud Speech-to-Text

Nécessite de créer un projet Google Cloud et en plus c'est payant (0.24€ par minute), donc je n'ai pas pu le tester.

Permet d'avoir une spatialité temporelle des phrases.

6.2 Gestion et extraction d'informations intégrées dans les fichiers vidéo

Modes de stockage des sous-titres

Les sous-titres peuvent directement gravés sur la vidéo donc on ne peut pas les extraites en tant que données car ils font partie de la vidéo elle-même.

Sous-titres multiplexés, dans ce cas ils sont inclus en tant que piste distinctes dans le conteneur vidéo (le plus souvent dans des formats comme MKV, MP4 ou AVI). Ces sous-titres sont souvent au format STR, ASS ou SUB.

Enfin les sous-titres peuvent être stockés séparément dans les mêmes formats que cité précédemment. C'est seulement du texte synchronisé avec la vidéo.

Formats de Sous-Titres

SRT (SubRip Subtitle) : Texte brut avec des timestamps. Simple et largement utilisé.

ASS (Advanced SubStation Alpha) : Format plus avancé, permettant des styles, des couleurs et des animations.

SUB/IDX : Images de sous-titres (bitmap) avec un fichier d'indexation.

Extraction

VLC comporte un système capable d'extraire les sous-titres et les exporter dans un fichier à part, mais il reste quand même assez limité dans sa prise en charge.

FFmpeg : outil qui permet d'une part d'afficher les métadonnées contenu dans le fichier vidéo (Principalement MKV et MP4).

```
Stream #0:0: Video: h264  
Stream #0:1: Audio: aac  
Stream #0:2: Subtitle: subrip
```

Figure 8: exemple de sortie

Mais aussi d'extraire des sous-titres et de le stocker dans un '.srt'.

6.3 Reconnaissance de texte dans les vidéos : Exploitation d'OCR (Reconnaissance Optique de Caractères)

Le moteur d'OCR **Tesseract** qui est développé par Google est un programme autonome écrit en C++. Il existe *pytesseract* qui est un wrapper Python et qui permet de l'utiliser dans un script. Il est rapide mais pas toujours précis.

EasyOCR est une bibliothèque OCR basée sur des CNN et des Transformers développée par JaideAI. Il supporte plus de 80 langues et il est beaucoup plus précis que Tesseract.

Action sociale

Le Département du Val-de-Marne mène une politique d'accueil et de soutien auprès de ceux qui rencontrent des difficultés sociales. Il assure la mise en place et le financement de dispositifs pour favoriser la lutte contre l'exclusion et réduire les inégalités.

Figure 9: premier test effectué

Pour le premier test les deux moteurs sont capable d'extraire le texte de façon correcte.



Figure 10: image où le texte est plus dur à identifier

Pour la deuxième image voici le résultat obtenu pour Tesseract :

```
Texte extrait:
La vie est trop
CN ea
ANAL
A UN

N {} -
```

Figure 11: texte extrait par Tesseract sur l'image 2

Le résultat pour EasyOCR est bien meilleur :

[**'La vie est'**, **'courte pour être'**, **'autre chose'**, **'quheureux'**, **'trop'**]

Sur la dernière image Tesseract n'extrait aucun texte tandis que EasyOCR arrive à extraire ce texte :

[**'SABINES'**, **'Montpellicr'**, **'Tomeration'**]

Il existe aussi MMOCR qui est utilisée du Deep Learning et qui est censé avoir une meilleure précision mais que je n'ai pas testé.

6.4 Segmentations de scènes

6.4.1 Shot Boundary Detection (SBD)

Différents types de transition

Cut Transition Un changement instantané d'une image à une autre sans effet intermédiaire. C'est la transition la plus rapide et la plus courante dans le montage vidéo.

Fade out L'image s'assombrit progressivement jusqu'à devenir complètement noire (ou d'une autre couleur). Cette transition est souvent utilisée pour marquer la fin d'une séquence ou d'une scène.

Fade in L'inverse du Fade Out : une image noire devient progressivement visible. Elle est souvent utilisée pour introduire une nouvelle scène ou séquence.

Dissolve Une image s'estompe progressivement pendant qu'une autre apparaît en surimpression. Les deux images se chevauchent pendant une courte période. Cette transition est souvent utilisée pour montrer un passage dans le temps ou un lien narratif entre deux scènes.

Wipe Transition Une image est remplacée par une autre selon un mouvement spécifique (ligne, courbe, motif). Contrairement au Dissolve, les deux images ne se mélangent pas mais sont séparées par une bordure en mouvement. Il existe différentes variantes qui sont les suivantes :

Motion from bottom to top, Motion from right to left, Wipe with oblique motion, The movement is going to the center.

Différentes méthodes pour détecter ces transitions

Méthode basée sur les pixels Cette méthode compare soit pixel par pixel ou bien le pourcentage de pixel changé et si ça dépasse un certain seuil fixé alors on parle de changement de plan. Bien que simple, cette méthode est très sensible aux variations mineures tel que les changements d'éclairage ou les diverses mouvements de caméra.

Méthodes basées histogramme En comparant les histogrammes ces méthodes détectent des variations globales entre les images. Il existe différentes méthodes de comparaison d'histogramme qui sont utilisé, il y a la différence d'histogramme global, l'intersection des histogrammes et la distance de Bhattacharyya.

Ces méthodes vont être plus robuste que celle basée sur les pixels mais par contre elles vont être inefficace lorsque les scènes auront des couleurs similaires et sur des grands mouvements de caméra rapide.

Méthodes basées sur les transformations Ces approches utilisent des transformations telles que la transformée en cosinus discrète (DCT) ou la transformée en ondelettes pour extraire des caractéristiques fréquentielles des images, facilitant la détection des transitions.

Ces méthodes vont être beaucoup plus robuste par contre plus complexe à calculer et elle nécessite un seuil qui soit finement ajusté.

Méthodes basées sur les contours En se concentrant sur les informations de contour ou de bord des images, ces techniques détectent des changements structurels significatifs entre les plans. Avec notamment l'extraction de bords par Sobel ou Canny on peut mesurer la variations des contours pour détecter une transition. Ces méthodes vont être très précise pour les changements de structures mais elle seront moins efficace pour les transitions douces.

Méthodes basées sur le mouvement Ces méthodes analysent le flux optique ou les vecteurs de mouvement pour distinguer les mouvements internes au plan des véritables transitions entre les plans. Une grande variation du flux optique peut signaler un changement de scène. Très utile pour détecter les transitions en présence de mouvement de caméra par contre ces méthodes nécessite des calculs complexes et peuvent confondre les mouvements rapides avec des changements de scène.

Méthodes basées sur l'apprentissage automatique

- Algorithmes de classification supervisés : Utilisent des features comme l'histogramme, les bords, et les vecteurs de mouvement. Puis un modèle tel que SVM est entraîné à reconnaître les transitions.
- CNN : Utilise une grande quantité de données annotées pour entraîner ce modèle.

Ces méthodes vont être très efficace pour détecter différents types de transitions par contre elles vont nécessiter un grand volume de données annotés.

Il est possible de combiner ces différentes méthodes pour obtenir des meilleurs résultats. On peut utiliser les méthodes histogrammes et contours pour détecter les cuts, mais aussi le flux optique et CNN pour identifier de manière optimal les transitions les plus complexes.

6.5 PySceneDetect

PySceneDetect est une bibliothèque Python permettant de détecter automatiquement les changements de plan dans une vidéo et de la segmenter en clips distincts. Elle a notamment été utilisée pour la segmentation de scènes dans l'application Celluloid.

Elle propose plusieurs algorithmes de détection, chacun adapté à des cas d'usage spécifiques :

- Content Detector : Identifie les coupes en analysant les variations de couleur et d'intensité entre les images successives. La différence est mesurée dans l'espace colorimétrique HSV et comparée à un seuil défini pour détecter les transitions brusques.
- Adaptive Detector : Fonctionne en deux passes. Il commence par calculer un score de changement d'image à l'aide du Content Detector, puis applique une moyenne mobile pour lisser les résultats. Cette approche permet de réduire les fausses détections, notamment en présence de mouvements de caméra.
- Histogram Detector : Compare la différence entre les histogrammes du canal Y (luminosité) dans l'espace colorimétrique YUV des images adjacentes, ce qui permet d'identifier les coupures basées sur des variations globales de luminosité.
- Hash Detector : Utilise un algorithme de hachage perceptuel pour détecter les changements de plan. Il applique une transformation en cosinus discrète (DCT) suivie d'un filtrage passe-bas, puis utilise un seuil basé sur la médiane pour identifier les transitions.

Il est également possible de combiner plusieurs algorithmes afin d'améliorer la précision de la détection. Cependant, cette approche entraîne une augmentation du temps de calcul et du nombre de faux positifs, nécessitant ainsi un ajustement des seuils et des paramètres pour optimiser les résultats.

Une multitude d'options sont disponibles, dont une particulièrement intéressante qui permet de réduire la qualité du média en entrée. Cela peut être utile pour diminuer le temps de calcul lorsque les fichiers sources sont en haute résolution (1080p, 2K

ou 4K), tout en conservant une détection efficace.

Pour plus de détails, vous pouvez consulter la documentation officielle : PySceneDetect API.

J'ai effectué des tests pour comparer les performances et les temps de calcul des différents algorithmes :

Vidéo couleur (20 secondes, bonne qualité) : Tous les algorithmes ont détecté l'ensemble des coupes. Les temps de calcul sont similaires entre eux.

Extrait de clip vidéo (1 minute, noir et blanc, bonne qualité) : 24 cuts étaient présents à détecter.

Algo	Vrai Positif	Faux Positif	Précision	Rappel	F1 Score
Content	17	1	94.44%	70.83%	80.95%
Adaptive	19	2	90.48%	79.17%	84.44%
Hash	20	2	90.91%	83.33%	86.96%
Histogram	23	4	85.19%	95.83%	90.20%

Table 2: Comparaison des résultats des algorithmes

Film d'Hitchcock (1h 41m 34s) : Mesure des temps de calcul pour chaque algorithme sur une vidéo longue.

Algo	Temps de calcul
Content	01'22"48
Adaptive	01'11"00
Hash	01'08"46
Histogram	00'57"42

Table 3: Comparaison des temps de calculs des algorithmes

6.6 YOLO : You Only Look Once

La vitesse et la précision de YOLO en font un choix judicieux pour les tâches de détection d'objets en temps réel dans diverses applications.

6.6.1 Détection d'objets et identification

On peut l'utiliser pour détecter différent objets dans l'image. En chargeant des poids pré-entraînés on peut aussi identifier un certains nombres d'objets.

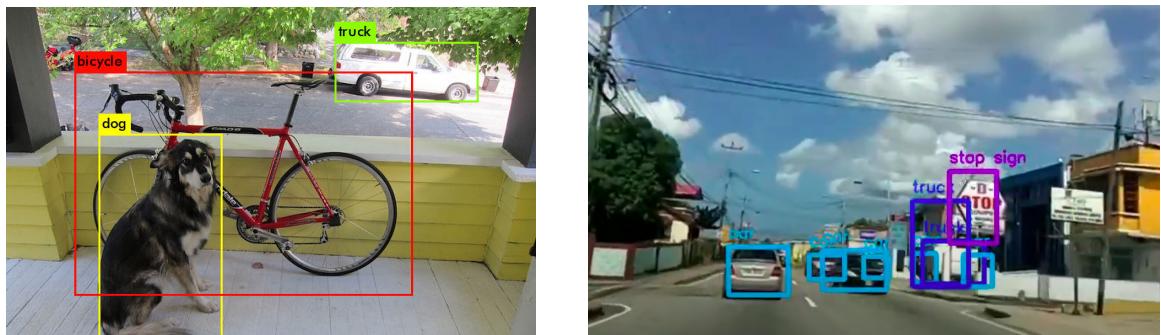


Figure 12: exemple d'utilisation

6.6.2 Poursuite de cible

De la même manière que précédemment pour la détection on peut aussi s'en servir pour suivre une ou plusieurs cible dans des images ou une vidéo.

6.7 Détection de Visage

DeepFace est un système de reconnaissance faciale basé sur l'apprentissage profond créé par un groupe de recherche de Facebook. Il identifie les visages humains dans les images numériques . Le programme utilise un réseau neuronal à neuf couches avec plus de 120 millions de poids de connexion et a été formé sur quatre millions d'images téléchargées par les utilisateurs de Facebook.

Cette librairie python propose différentes fonctionnalités qui sont les suivantes :

Face Verification A partir de deux images DeepFace est capable de terminer si oui ou non la personne de l'image 1 et bien celle de l'image 2 en extrayant de chaque image le visage. De base il utilisera la distance cosinus pour comparer les deux vecteurs caractéristiques extraits de chaque image respectivement, mais on a aussi la possibilité de faire la comparaison avec la distance euclidienne ou bien la distance euclidienne normalisée.

Face Recognition De la même manière que précédemment il est capable d'identifier si la ou les personnes présente dans une image appartient à la base de données.

Embeddings On peut aussi calculer le vecteur caractéristiques d'une image. La taille de ce vecteur peut varier en fonction du modèle utilisé, par exemple pour le modèle VGG-Face sa taille est de 4096.

Face recognition models DeepFace propose différents modèles, voici les plus populaires avec leurs précisions (calculé et non celle déclarée par les développeurs):

- VGG-Face (celui qui est utilisé de base) : Précision → 96.7%
- Google FaceNet : Précision → 97.4%
- OpenFace : Précision → 78.7%
- Facebook DeepFace : Précision → 69.0%
- DeepID : Précision → 66.5%
- Dlib : Précision → 96.8%
- ArcFace : Précision → 96.7%

Facial Attribute Analysis DeepFace a aussi un puissant module d'analyse des attributs du visage comprenant **l'âge**, **le sexe**, **l'expression faciale** (avec ces différentes expressions faciales possibles : la colère, la peur, le neutre, la tristesse, le dégoût, la joie et la surprise) et **la race** (asiatique, blanc, moyen-oriental, indien, latino et noir)



Figure 13: exemple de résultats pour l’analyse du visage

Face Detection and Alignment

Alignement Il permet de garantir que toutes les images de visages sont dans une position cohérente, ce qui facilite des tâches comme la reconnaissance faciale, l’extraction de caractéristiques, ou la classification. Il est donc possible de choisir si on applique l’alignement ou non.

Détection de visage Il existe aussi différentes variantes de détection de visage :

- OpenCV (celui qui est utilisé de base) : c’est le détecteur de visage le plus léger. Il utilise un algorithme haar-cascade qui n’est pas basé sur des techniques d’apprentissage en profondeur. C’est pourquoi il est rapide, mais ses performances sont relativement faibles
- Dlib : Ce détecteur utilise un algorithme HOG, comme OpenCV il n’est pas basé sur l’apprentissage profond. Néanmoins, il présente des scores de détection et d’alignement relativement élevés.
- SSD : SSD signifie Single-Shot Detector ; il s’agit d’un détecteur populaire basé sur l’apprentissage profond. Les performances du SSD sont comparables à celles d’OpenCV. Cependant, le SSD ne prend pas en charge les repères faciaux et dépend du module de détection oculaire d’OpenCV pour s’aligner. Même si ses performances de détection sont élevées, le score d’alignement n’est que moyen.
- MTCNN : Il s’agit d’un détecteur de visage basé sur l’apprentissage profond et livré avec des repères faciaux. C’est la raison pour laquelle les scores de détection et d’alignement sont élevés pour MTCNN. Cependant, il est plus lent qu’OpenCV, SSD et Dlib.
- RetinaFace : RetinaFace est reconnu comme étant le modèle de pointe basé sur l’apprentissage profond pour la détection des visages. Cependant, cela nécessite une puissance de calcul élevée. C’est pourquoi RetinaFace est le détecteur de visage le plus lent par rapport aux autres.

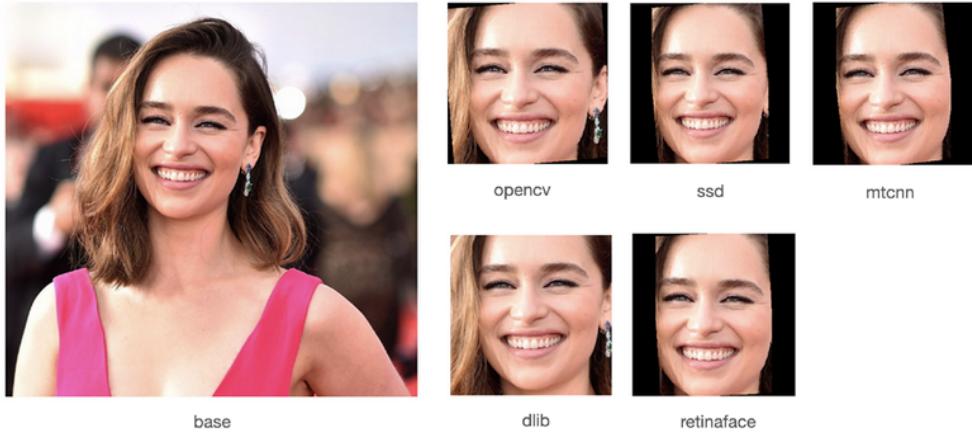


Figure 14: exemple de résultats pour la détection du visage

6.8 Optimisation des Performances

Dans le cadre du développement d'un logiciel de traitement vidéo avec Python et Qt, l'objectif est d'offrir une expérience utilisateur à la fois intuitive et réactive. Il est essentiel que l'utilisateur puisse exécuter ses tâches avec un minimum de complexité tout en obtenant des résultats rapidement. Cependant, les différentes méthodes d'analyse et de détection des changements de scène impliquent des traitements intensifs en calcul, pouvant entraîner des temps de réponse élevés, surtout sur des vidéos haute résolution.

Afin de garantir des performances optimales, l'optimisation des calculs devient un enjeu majeur. L'intégration de CUDA via PyCUDA ou CuPy permet d'exploiter la puissance du GPU pour exécuter des opérations massivement parallèles, réduisant ainsi considérablement le temps de calcul par rapport aux traitements effectués uniquement sur CPU.

Toutefois, dans un logiciel basé sur Qt (via PyQt ou PySide), l'optimisation ne se limite pas à l'accélération des calculs. Il est crucial de garantir la fluidité de l'interface utilisateur en évitant les blocages dus à des traitements lourds. Pour cela, l'exécution des tâches CUDA sera déléguée à des threads ou des tâches asynchrones, permettant ainsi à l'interface de rester réactive pendant le traitement vidéo.

Opération	CPU	GPU	Gain
Filtre Gaussien (1920*1080)	120ms	5ms	24 fois plus rapide
Détection de contours (Canny)	200ms	10ms	20 fois plus rapide
Reconnaissance faciale (DeepFace)	500ms	40ms	12 fois plus rapide

Table 4: Exemple de gain de temps

7 Choix des langages de programmation

7.1 Python

Écosystème riche en bibliothèques spécialisées Python dispose d'un grand nombre de bibliothèques utile dans le traitement vidéo (beaucoup ont été présenté précédemment).

- OpenCV pour le traitements d’images et de vidéos, la détection de contours, suivi d’objets etc ...
- FFmpeg pour la manipulation et la conversion de fichier vidéo.
- Tenserflow pour l’intégration de modèle d’apprentissage profond.

Interopérabilité* avec C++ et CUDA Python peut facilement appeler du code C++ (via Cython, Pybind11), ce qui permet d’intégrer des parties critiques en C++ pour des gains de performances.

Il est compatible avec CUDA via PyCUDA ou CuPy, permettant l’accélération GPU sans changer de langage.

Portabilité et compatibilité multi-plateforme Python fonctionne sur Windows, Linux et macOS, ce qui simplifie le déploiement.

7.2 Qt

L’interface utilisateur joue un rôle clé dans un logiciel de traitement vidéo. Qt est l’un des frameworks les plus puissants pour concevoir des interfaces graphiques en Python.

Interface moderne et flexible Qt permet de créer des interfaces complexes avec des composants avancés : lecteurs vidéo, calques, annotations, etc. Supporte le multi-fenêtrage, essentiel pour comparer plusieurs vidéos côté à côté.

Compatibilité avec OpenGL et GPU Qt permet l’intégration d’OpenGL pour accélérer l’affichage vidéo. Compatible avec CUDA pour tirer parti du GPU dans le rendu en temps réel.

Gestion avancée des événements et du threading Qt intègre un système de gestion des threads (QThread, QRunnable) qui permet d’exécuter des tâches lourdes en arrière-plan sans bloquer l’interface. Utile pour éviter que le traitement vidéo ne bloque l’interaction utilisateur.

8 Définition

8.1 Étude iconographique

Étude descriptive des différentes représentations figurées d’un même sujet ; ensemble classé des images correspondantes.

8.2 Profilmique

Tout ce qui existe dans la réalité et qui pourra être enregistré par la caméra pour les besoins du film.

8.3 Interopérabilité

C’est la capacité que possède un produit ou un système, dont les interfaces sont intégralement connues, à fonctionner avec d’autres produits ou systèmes existants ou futurs et ce sans restriction d'accès ou de mise en œuvre

9 Références

- 3.1VLC Media Player - Page Wikipédia
- 3.2QuickTime - Page Wikipédia
- 3.3Screen - Site Officiel
- 3.4IINA - Site Officiel
- 3.5MPV - AlternativeTo
- 3.6DaVinci Resolve - AlternativeTo
- Comparaison des différents lecteurs
- 4.1Ronfard R. & Valero J., "KinoAI et le carnet audiovisuel : une solution plurielle pour l'étude des répétitions", European Journal of Theatre and Performance, mai 2020.
- 4.1KinoAI
- 5.1inaSpeechSegmenter - Github
- 5.2inaFaceAnalyser - Github
- 5.3speaking-clock-detection - Github
- 6.4.1Benoughidene A. & Titouna F., Shot Boundary Detection: Fundamental Concepts and Survey
- 6.7DeepFace - Documentation
- 6.7DeepFace - viso.ai