Méthodes de vision par ordinateur pour la segmentation, l'analyse de pose et la réinsertion vidéo

Tom LE BER Tony PEROTTINO

TER L3 MIDL – Université de Toulouse

Table des matières

Préambule

Ce travail a été réalisé dans le cadre de la Double Licence Mathématiques-Informatique (MIDL) de l'université de Toulouse. Il correspond à la matière "Stage MIDL 2" consistant en un travail dirigé de recherche (TER) dans les domaines de l'informatique et/ou des mathématiques. Ce rapport fait office de compte-rendu de l'entièreté de notre travail de recherche.

1 Introduction

Ce TER s'inscrit dans une démarche plus générale, qui est de déterminer analytiquement la présence de montages par IA, mieux connus sous le nom de deepfakes.

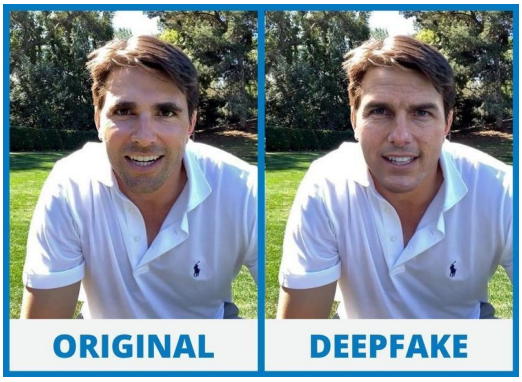

Figure 1 – Exemple de deepfake (réinsertion de visage)

Figure 2 – Exemple de deepfake (compositions d'élements)

Le deepfake est une technique de manipulation numérique utilisant l'IA pour créer des images, des vidéos ou des audios hyperréalistes mais en réalité entièrement factices. Ce procédé permet de simuler des actions ou des paroles n'ayant jamais eu lieu, entraînant entre autres des risques majeurs de désinformation, d'usurpation d'identité et de manipulation de l'opinion.

Plutôt que de chercher à repérer directement les manipulations finales, l'objectif de ce TER est d'étudier un procédé courant dans la manipulation vidéo : la segmentation vidéo. Nous proposons dans ce rapport la mise en place d'une chaîne de traitement permettant d'analyser une vidéo de danse, d'isoler automatiquement la personne présente dans la scène, puis de la réinsérer sur un nouveau fond de la manière la plus réaliste possible. Ce problème mobilise plusieurs thématiques de vision par ordinateur à savoir l'analyse de pose, la segmentation vidéo, la reconstruction d'une vidéo détourée et l'évaluation qualitative et quantitative des résultats obtenus.

La segmentation vidéo est une méthode de traitement d'image qui consiste à diviser chaque image d'une séquence en plusieurs régions ou segments homogènes. Les critères peuvent être basés sur la forme, la couleur, un mouvement, ou même de traquer un objet ou une personne selon ce qu'elle est.

Dans le contexte du deepfake, la segmentation vidéo est un scalpel numérique permettant d'isoler précisément les éléments que l'IA devra modifier ou remplacer. C'est ce pré-traitement qui crée alors l'illusion de réel, puisque les vidéos sans retouches sont systématiquement basées sur la réalité dans les deepfakes les plus réussis.

Dans notre cas d'étude, nous avons utilisé un ensemble de vidéos de pole dance dans l'objectif de traquer exclusivement le danseur (ou la danceuse). Il s'agit d'une tâche difficile car la segmentation doit comprendre que l'objet à découper ne comprend pas la barre de pole, qui passe périodiquement à l'avant et l'arrière du corps. L'objectif est donc de produire un masque correct sur une vidéo complète, et de conserver une cohérence temporelle sur l'ensemble de la séquence.

La sélection de vidéo de pole dance est aussi utile car les membres et articulations à la jonction de la barre sont peu mobiles et peuvent être appliqués à d'autres usages dans le domaine de l'armée (non abordé dans la suite du rapport). Nous prendrons le temps de décrire à travers le rapport les nombreux points de difficultés plus en détails.

2 Problématique et objectifs

Le problème étudié peut être résumé de la manière suivante : étant donnée une vidéo contenant une danseuse, comment détourer automatiquement et de manière suffisamment précise pour permettre une réinsertion convaincante sur un nouveau fond ? Cette formulation, simple en apparence, recouvre en réalité plusieurs sous-problèmes.

— Utilisation d'outil : étudier plusieurs outils d'analyse de pose et de segmentation afin de comparer leurs apports et leurs limites.

— Segmentation : mettre en place un pipeline fonctionnel capable d'isoler automatiquement la danseuse sur une séquence vidéo.

— Evaluation de la qualité : proposer des critères d'évaluation, visuels et quantitatifs, permettant de juger la qualité de la reconstruction vidéo produite.

Le premier enjeu réside dans l'arbitrage entre la performance du résultat et le coût computationnel des outils d'analyse de pose. En effet l'enjeu est à la fois technique : réaliser la jonction entre différents outils déjà existants (nous étudierons notamment `MMPose`, `NLF` et différentes versions de Segment Anything aussi dit `SAM`) ; mais aussi méthodologique : quels enchaînement d'outils permettent le meilleur ratio temps/performance ?

Le second point est la stratégie d'utilisation de ces outils pour créer une segmentation fiable et cohérente de la vidéo originale. Ce critère est en phase avec la méthodologie, de mauvaises performances ou la découverte de nouveau outils reconduisent à la première tâche. Il est alors primordial de trouver les meilleurs compromis pour extraire les meilleurs calques.

Enfin, la réinsertion finale et son évaluation constituent l'aboutissement de la pipeline. La difficulté est ici de mesurer l'efficacité du détourage non seulement par la précision du masque, mais aussi par la fluidité de l'intégration sur le nouveau fond. L'oeil humain doit à l'issue de la réinsertion être idéalement dupé, la nouvelle vidéo se doit d'être convaincante. Il est alors possible d'extraire des données de cette reconstruction afin de la catégoriser d'un score (par exemple). Ce dernier sera aussi un enjeu de ce TER.

3 Évolution du travail et présentation des versions

Nous avons choisi de réaliser nos notebooks sur Google Colab, car ils permettent une execution identique, peu importe la machine utilisée (via le cloud). Notre travail pourra alors être facilement réutilisé, complété et amélioré par d'autres chercheurs et stagiaires.

Le travail réalisé s'est structuré autour de quatre versions principales d'un notebook Google Colab. Ces notebooks ne correspondent pas simplement à des copies successives d'un même code : ils reflètent une évolution progressive de la méthodologie et de la manière d'aborder le problème, comme explicité dans l'introduction. Chaque version représente une amélioration notable et en rupture par rapport à leur précédente.

3.1 Version V1 : étude de `NLF`

La première version du notebook est centrée une étude comparative de `NLF` par rapport à `MMPose`. Ces deux outils réalisent les mêmes objectifs : trouver les points d'un squelette sur des silhouttes animales, humaines, ou de mains. Ici seulement les modèles sur les humains nous intéressent. L'étude de `MMPose` ayant été rapidement écarté, il ne reste plus de mention dans ce rapport, étant donné la fiabilité de `NLF`.

L'objectif principal est donc d'évaluer la capacité du modèle de `NLF` à détecter la personne présente dans la vidéo et à fournir une estimation de pose 2D/3D exploitable. Cette première étape a permis de construire des visualisations simples, de mesurer le temps d'inférence et de mieux comprendre le type d'information que l'analyse de pose pouvait apporter au reste du pipeline.

3.2 Version V2 : intégration de `SAM 2`

La deuxième version introduit un changement important : l'ajout de `SAM 2` pour la segmentation. L'idée est d'utiliser les résultats de l'analyse de pose comme points d'ancrage, puis de propager les masques dans la vidéo grâce à l'outil `SAM 2`. Cette version constitue le premier véritable pipeline de segmentation et de reconstruction de la séquence détourée. Elle introduit également les premières réflexions sur la réinsertion et sur les limites concrètes du détouré obtenu, notamment au niveau des cheveux ou des parties fines du corps.

3.3 Version V3 : pipeline principal et évaluation des résultats

La version `V3` correspond à la version la plus aboutie utilisant les outils `NLF` et `SAM 2`. Elle reprend la logique de la `V2`, mais elle y ajoute un travail plus systématique sur l'analyse des résultats par la de recherche de métriques justifiant la qualité de reconstruction vidéo, en plus d'une possibilité de tester n'importe quelle vidéo hors du dataset d'origine. C'est sur cette version dont le rapport s'appuie le plus, et comme base des différentes implémentations proposées dans le projet.

3.4 Version V4 : pipeline de segmentation par `SAM 3`

La version `V4` correspond à une implémentation fonctionnelle de segmentation vidéo reposant entièrement sur `SAM 3`, car cette version est plus puissante et permissive. Contrairement aux versions `V2` et `V3`, elle n'a plus besoin de `NLF` pour fournir des joints 2D servant à initialiser la segmentation. L'approche devient alors plus directe : un prompt texte est d'abord sélectionné sur une frame de référence, puis `SAM 3` est utilisé comme unique moteur de segmentation et de propagation vidéo.

L'intérêt principal de cette version est qualitatif : les calques obtenus sont globalement plus cohérents que dans la `V3`, en particulier sur les parties fines du sujet comme les cheveux, qui étaient fréquemment dégradés dans l'approche `NLF + SAM 2`. En contrepartie, le coût computationnel augmente fortement. La `V4` doit donc être comprise comme une alternative plus lente mais qualitativement plus solide pour la segmentation vidéo.

4 Données et protocole expérimental

Les données dont nous avons à disposition pour ce TER est un ensemble d'environ 50 vidéos de pole dance. Chaque vidéo est préalablement découpée en frames individuelles puis traitée par les différents modules du pipeline.

La majorité de nos expérimentations seront faites sur une vidéo principale, elle est en quelque sorte notre vidéo témoin. La séquence étudiée contient 1329 frames et un ensemble de mouvements diversifié avec différentes vitesses, ce qui fournit une base suffisamment longue pour évaluer à la fois la qualité instantanée de la segmentation et sa stabilité temporelle.

Le protocole général adopté au cours du stage peut être résumé comme suit, toutes ces étapes sont effectivement présentes à partir de la `V3` :

1. Charger la vidéo et extraire l'ensemble des frames
2. Appliquer un module d'analyse de pose pour obtenir des informations géométriques sur la personne
3. Utiliser ces informations pour initialiser ou guider une méthode de segmentation vidéo
4. Produire une vidéo détourée sur fond noir
5. Réinsérer la personne détourée sur un nouveau fond
6. Evaluer qualitativement et quantitativement les résultats obtenus

L'étape numéro 3 varie en fonction de la version du notebook utilisée et est détaillé dans la suite du rapport.

5 Analyse de pose : pistes explorées et choix retenus

L'analyse de pose constitue une première brique importante du projet. Elle permet de repérer la danseuse dans l'image, de visualiser la structure du corps et, dans certaines configurations, de fournir des informations utilisables pour l'initialisation de la segmentation.

5.1 Perspectives explorées

Au cours du stage, plusieurs pistes ont été envisagées pour cette étape. Parmi elles, `MMPose` a été considéré comme une piste possible pour l'estimation de pose, notamment dans une optique de comparaison avec `NLF`. Toutefois, seulement `NLF` sera utilisé dans l'ensemble des notebooks pour l'analyse de pose, car plus fiable et plus rapide. Cette approche a été privilégiée car elle permettait d'obtenir rapidement des joints (points représentant le squellette de la personne étudiée) 2D et 3D exploitables et de les intégrer à la suite du pipeline.

5.2 Résultats obtenus avec `NLF`

La version `V1` du notebook a permis de mener une première étude détaillée de `NLF`. Sur la vidéo principale, 1329 frames ont été traitées. Le notebook indique qu'une personne est détectée sur 1263 frames, soit un taux de détection d'environ 95%. Le temps moyen d'inférence observé est de 309,7 ms par frame, correspondant à un débit d'environ 3,23 images par seconde.

Ces résultats ont montré que `NLF` constitue un bon outil d'analyse de pose pour obtenir une compréhension grossière du mouvement de la danseuse au cours du temps (étant donné que le résultat est un ensemble de points du squelette). Les visualisations produites dans le notebook permettent notamment d'observer :

— un overlay 2D des joints sur une frame,

Figure 3 – Exemple de détection 2D produite par `NLF` sur une frame de la vidéo principale.

— une mosaïque de plusieurs frames annotées,

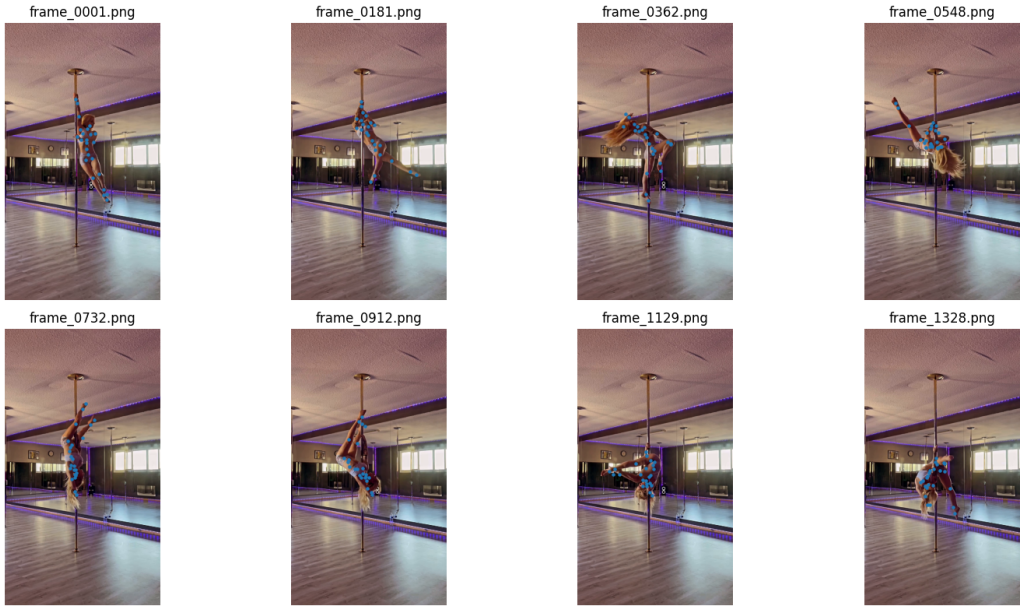

Figure 4 – Mosaïque de plusieurs frames annotées par `NLF`. Cette visualisation permet d'apprécier qualitativement la robustesse du modèle sur différentes postures.

— l'évolution temporelle de la coordonnée Z d'un joint,

— une visualisation 3D de la pose estimée.

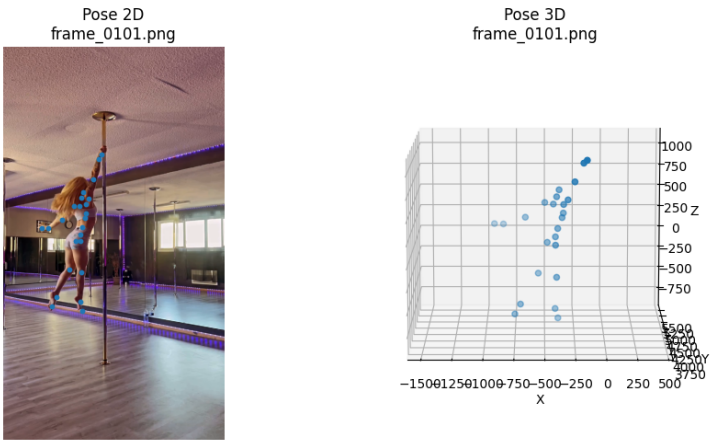

Figure 5 – Exemple de visualisation 2D/3D de la pose estimée.

5.3 Limites de l'analyse de pose

Même si `NLF` fournit des résultats utiles, cette étape présente plusieurs limites. D'une part, une estimation de pose correcte ne garantit pas une bonne segmentation : disposer de joints ne suffit pas à reconstruire précisément les contours du corps, notamment pour les cheveux, les vêtements ou les membres partiellement flous. D'autre part, la qualité des prédictions peut varier selon la posture, l'occlusion par la barre ou la dynamique du mouvement.

Cette limite devient particulièrement visible lorsqu'on compare la `V3` à la `V4`. Dans la `V4`, l'analyse de pose n'est plus utilisée pour la segmentation : `SAM 3` segmente directement la danseuse à partir d'un prompt texte, sans passer par une étape préalable de joints 2D/3D. Cette modification simplifie conceptuellement la pipeline de découpe et permet de mieux préserver certaines structures fines du sujet, mais au prix d'un temps d'exécution beaucoup plus élevé, de l'ordre de 1h30 sur la vidéo témoin.

6 Segmentation vidéo et reconstruction : deux choix d'implémentation (V3 & V4)

La segmentation vidéo de la `V3` repose sur l'utilisation conjointe de `NLF` et de `SAM 2`. `SAM` permet de détourer des formes, notamment de personnes, sur un flux photo ou vidéo. Cet outil prend en entrée un ensemble de points à l'intérieur et à l'extérieur de la forme désignée, puis produit un calque binaire de la région trouvée. Dans la `V3`, la contribution centrale du projet consiste alors à utiliser les joints 2D fournis par `NLF` pour guider périodiquement `SAM 2` sur la vidéo.

Une seconde implémentation existe désormais dans la `V4`. Elle remplace totalement cette stratégie par une segmentation vidéo entièrement basée sur `SAM 3`, pilotée à partir d'un prompt texte et propagée directement sur la séquence. Cette alternative ne nécessite plus `NLF`, et elle donne sur la vidéo témoin des calques globalement plus cohérents, mais pour un coût de calcul beaucoup plus important.

6.1 Stratégie du pipeline `NLF` + `SAM 2` (V3)

L'outil `SAM 2` utilisé seul sur notre vidéo témoin possède des résultats moyennement convaincants. La vidéo obtenue en pointant la danceuse sur la première frame manuellement puis en laissant la propagation s'effectuer sur le reste de la vidéo donne des calques clignotants et peu convaincants, à cause en partie des passages à répétition de la danceuse passant devant et derrière la barre de pole faussant la détection. On peut assez facilement remarquer que `SAM 2` perd le contexte de la forme quand il est lancé seul, il réussit à garder une forme correcte environ une dizaine de secondes.

La sortie de `NLF`, c'est-à-dire l'ensemble de joints 2D trouvés, nous permet de contourner ce problème : en les confiant à `SAM 2` pour qu'il dessine un forme à partir de ceux-ci, les résultats sont plus stables. En termes de temps d'exécution, `NLF` et `SAM 2` sont raisonnables, pour `NLF` il faut environ 200ms pour 1 frame (5 frames par secondes), donc $4'43''$ sur 1329 frames pour la vidéo témoin. Quant à lui, `SAM 2` est 7 fois plus lent environ pour une frame, soit 0,6 frames par seconde. Ménager `SAM 2` permet de gagner du temps.

On peut donc considérer la situation comme un curseur qui oscille entre l'usage de `NLF` et `SAM 2` dont les critères sont la qualité des calques et le temps d'exécution. La stratégie est d'utiliser périodiquement les deux outils sur des segents de frames définis à l'avance. Sous-utiliser `NLF` rend les calques instables comme dit précédemment, mais le sur-utiliser fait aussi chuter la qualité de la vidéo, car certaines positions où les joints superposent davantage la barre rend un calque combinant la danseuse à la barre. De plus `SAM 2` réagit moins bien quand la série de frame est trop courte. `NLF` a tendance à faire apparaitre la barre quand la segmentation est recalculée. Nous avons finalement opté pour un compromis : la vidéo est segmentée par morceaux successifs d'environ 2 secondes, chaque morceau étant initialisé à partir d'une frame d'ancrage puis propagé au moyen de `SAM 2`. Plus le découpage est fin, plus `NLF` prend aussi plus de temps à se lancer et s'arrêter. Au niveau du temps d'éxécution, on est autour de 17 minutes pour 1329 frames (0.767 frames par secondes), la durée pouvant atteindre un peu moins de 20 minutes avec `SAM 2` uniquement.

Cette approche présente deux intérêts principaux :

- séparer clairement le rôle de la pose et celui de la segmentation,
- permettre de relancer la segmentation régulièrement afin de limiter les dérives au cours du temps.

6.2 Résultats de segmentation `NLF` + `SAM 2` (V3)

Dans la version `V3`, l'inférence `NLF` est réalisée sur les 1329 frames de la séquence. Le temps moyen d'inférence indiqué pour ce cas particulier est de 176,8 ms par frame, soit environ 5,66 images par seconde. Le notebook signale également 1210 frames avec personne détectée dans ce cadre expérimental.

Une fois la segmentation effectuée, plusieurs visualisations permettent d'évaluer les résultats :

— le masque obtenu sur une frame unique avec son overlay

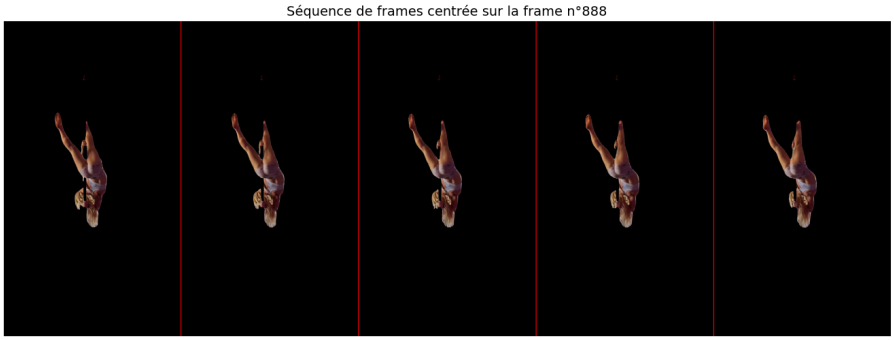

Figure 6 – Exemple de masque obtenu sur une frame de la vidéo par le pipeline principal `V3`.

— les anomalies de segmentation les plus marquées, détectées par les métriques temporelles

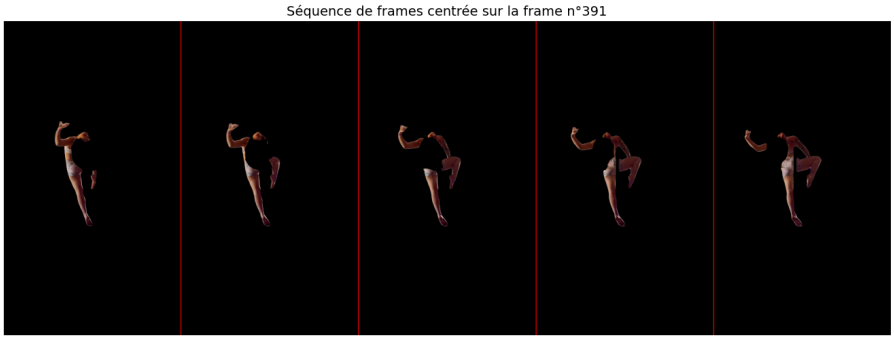

Figure 7 – Exemple d'anomalie de segmentation détectée automatiquement parmi les frames les plus incohérentes.

— la vidéo détourée produite sur fond noir.

6.3 Stratégie du pipeline `SAM 3` (V4)

La `V4` effectue la segmentation vidéo sans analyse de pose préalable (ex : les joints de `NLF` comme donnée d'entrée) et prend comme donnée d'entrée uniquement un prompt texte (celui ayant le meilleur taux de reconstruction parmi une série de prompts entrés). Ensuite, `SAM 3` détoure automatiquement l'objet étudié sur toute la vidéo, en procédant blocs par blocs de frames. Sur la vidéo témoin, cette approche supprime une grande partie des défauts visibles liés à la perte de contexte périodique de l'utilisation de `NLF` ou parce que `SAM 2` est moins évolué. La contrepartie de la qualité visuelle se retrouve dans le temps d'exécution, par exemple la vidéo témoin nécessite environ 1h30 pour 1329 frames.

Nous avons analysé plusieurs prompts différents parmi une sélection de 8 formulations personnalisées qui nous semblaient appropriées (`"woman doing pole dance"`, `"person near a pole"`, etc.).
Par exemple, le meilleur prompt sur notre vidéo témoin est `"person"`. La métrique de sélection du meilleur prompt correspond au taux de confiance de `SAM 3` délivré pour la segmentation de la première frame.

L'exécution complète de la segmentation est ensuite réalisée par morceaux successifs, avec un découpage en blocs de 8 frames et une propagation vidéo pilotée directement par `SAM 3`. Dans notre configuration Colab actuelle, cette approche demande environ 1h30 d'exécution sur la vidéo principale, soit environ une image toutes les 4 secondes.

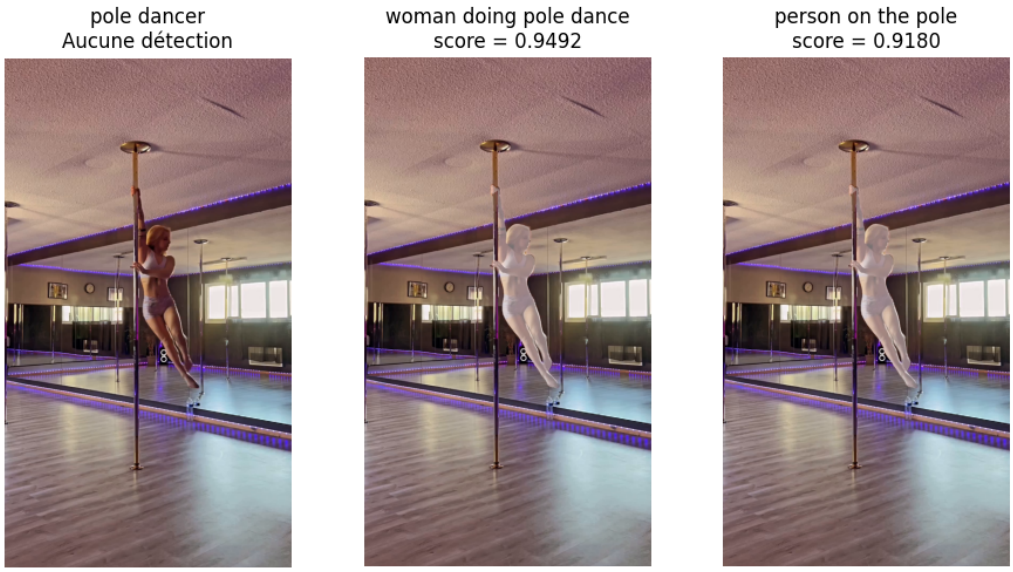

Figure 8 – Exemple de calibration d'un prompt texte dans la `V4` basée sur `SAM 3`.

6.4 Résultats de segmentation `SAM 3` (V4)

La `V4` a été exécutée sur la même vidéo témoin de 1329 frames que la `V3`, avec une segmentation entièrement pilotée par `SAM 3`. L'intérêt principal de cette version est qualitatif : les calques obtenus sont globalement plus complets et plus cohérents temporellement que dans la `V3`, en particulier sur les parties fines du sujet.

Sur la vidéo principale, l'amélioration la plus visible concerne la préservation de la silhouette supérieure de la danseuse, notamment la tête et les cheveux. Là où la `V3` perd régulièrement ces structures ou produit des calques partiellement disjoints, la `V4` conserve une forme plus continue du corps au fil du temps. Cette différence est particulièrement importante dans notre contexte, car les mouvements rapides, les postures extrêmes et les passages devant la barre rendent précisément ces zones difficiles à segmenter correctement.

La figure 9 illustre cette différence sur une même séquence de frames centrée sur la frame 391. Dans la `V3`, on observe une perte importante de la partie supérieure du corps, ainsi qu'une segmentation incomplète de la danseuse. Dans la `V4`, la silhouette reste nettement plus fidèle, avec une meilleure conservation des cheveux et de la continuité corporelle.

`V3 : NLF + SAM 2` `V4 : SAM 3`

Figure 9 – Comparaison qualitative entre la `V3` et la `V4` sur une séquence de frames centrée sur la frame 391. La `V4` conserve mieux la silhouette globale de la danseuse, en particulier la tête et les cheveux, alors que la `V3` produit ici une segmentation plus incomplète.

Sur la frame centrale 391, il est également possible de comparer directement les masques produits par les deux pipelines.

Figure 10 – Comparaison locale des masques obtenus sur la frame 391. Les zones blanches sont détectées par les deux pipelines, tandis que les zones rouges correspondent aux régions segmentées par la `V4` mais non détectées par la `V3`.

Cette amélioration qualitative se paie toutefois par un coût d'exécution nettement plus élevé. Là où la `V3` demande environ 17 minutes sur la vidéo témoin, la `V4` nécessite environ 1h30, soit un facteur d'environ 5 en temps de calcul. La `V4` apparaît donc comme une version plus convaincante visuellement pour la segmentation elle-même, tandis que la `V3` reste plus adaptée à des expérimentations rapides ou à la mise en place de métriques sur un grand nombre de séquences.

6.5 Limites qualitatives observées

Concernant la `V3`, malgré des résultats globalement encourageants, plusieurs défauts reviennent de manière récurrente. Le plus notable concerne les cheveux, qui ne sont pas toujours correctement intégrés au masque, du fait qu'ils ne soient pas inclus dans les points de `NLF` et que leur mouvement est plus rapide que le reste du corps se faisant entrainer dans les mouvements. De manière plus générale, les parties fines du corps et certains contours peuvent être mal segmentés, en particulier lorsque la posture est extrême ou lorsqu'une partie du corps est partiellement masquée par la barre.

Puisque le problème réside dans les transitions entre les deux outils, il aurait pu être envisagé de composer les fenêtres de frames de `SAM 2` l'une au-dessus de l'autre. Cela permettrait de fluidifier les bugs de transitions. Une fonction permettant de trouver le meilleur des deux mondes serait alors la question légitime, que nous avons choisi de ne pas creuser car les résultats étant déjà satisfaisants.

Concernant la `V4`, les résultats observés sont nettement plus convaincants sur la vidéo témoin. La propagation par `SAM 3` permet une meilleure préservation de la silhouette globale et des structures fines, là où la `V3` perdait régulièrement des détails comme les cheveux ou certaines extrémités. Cette amélioration reste toutefois conditionnée par un coût d'exécution beaucoup plus élevé, dépendant notamment de la qualité souhaitée pour la segmentation vidéo.

7 Réinsertion sur un nouveau fond

Une fois la vidéo détourée obtenue, la dernière étape du pipeline consiste à réinsérer la danseuse sur un nouveau fond. Dans le notebook, cette étape est réalisée à partir d'une transformation géométrique simple, définie à partir de deux points de référence correspondant à la barre sur la vidéo d'origine et sur l'image de destination.

7.1 Principe de la réinsertion

L'idée est de construire une transformation affine de similarité (rotation, mise à l'échelle, translation) telle que l'axe de la barre dans la vidéo d'origine soit aligné avec l'axe choisi sur l'image de fond. Une fois cette transformation calculée, chaque frame détourée peut être reprojetée et composée avec le nouveau fond.

Cette stratégie est suffisante pour produire une première démonstration fonctionnelle du pipeline. Elle présente toutefois une limite structurelle : elle repose sur un réglage manuel des points de correspondance et ne s'adapte pas automatiquement à des changements de scène plus complexes.

7.2 Intérêt et limites

L'intérêt principal de cette étape est de montrer que le détouré vidéo produit n'est pas seulement une fin en soi, mais qu'il peut s'inscrire dans une chaîne complète de transformation vidéo. Toutefois, la qualité de la réinsertion dépend fortement de la qualité des masques et du bon choix des points de recalage. En présence d'erreurs de segmentation ou de décalages géométriques, les artefacts deviennent rapidement visibles.

Figure 11 – Exemple de réinsertion de la danseuse sur un nouveau fond.

8 Évaluation quantitative des résultats

Un aspect important à partir de la version `V3` du projet est l'analyse consacré à l'évaluation quantitative des masques générés. Plutôt que de se limiter à une inspection visuelle, plusieurs métriques ont été mises en place pour détecter automatiquement des incohérences ou mesurer la stabilité temporelle du détouré obtenu.

8.1 Objectif des analyse

L'objectif principal de cette partie est de faire correspondre un calque vidéo (produit des parties précédentes) avec un score de reconstruction. Nous avons pensé et discuté plusieurs approches, dont nous pensons que la plus forte prédiction est une combinaison d'indicateurs. Chaque indicateur pourrait dans l'idéal permettre d'assurer qu'une frame est mal reconstruite, et le score final serait le ratio de l'ensemble des frames corrompues par rapport à l'ensemble des frames. Or nous avons rencontré des problématiques inattendues, qui remettent en question la qualité de nos indicateurs.

Ainsi nous allons regarder les pistes que nous avons explorées, bien qu'un score final n'ait pas été implenté, faute de qualité générale des indicateurs explorés.

8.2 Analyse des aires

Une première famille de métriques repose sur l'aire des masques. Pour chaque frame, on compte le nombre de pixels blancs dans le masque sous forme binaire, puis on étudie l'évolution de cette quantité au cours du temps. On s'est intéressée à la variation et l'accélération des aires, qui sont mathématiquement des taux de variations instantannées par rapport aux valeurs adjacentes.

$$V_n = \frac{A_n - A_{n-1}}{\Delta t} \tag{1}$$

$$\gamma_n = \frac{V_n - V_{n-1}}{\Delta t} = \frac{A_n - 2A_{n-1} + A_{n-2}}{(\Delta t)^2} \tag{2}$$

Une variation brutale de l'aire peut traduire un échec de segmentation : disparition partielle du corps, perte des cheveux, ou au contraire inclusion intempestive d'une partie du fond. Les valeurs les plus extrêmes sont souvent des problèmes réels, mais passé ces quelques frames (Moins de 0,5%) les frames qui ont des variations d'accélération brusques sont souvent correctes. Il est à noter que si la caméra bouge au fil de la vidéo, l'indicateur sur les aires sera très sensible et probablement inutilisable étant donné que le/la danceur couvera plus ou moins de pixels en fonction du déplacement. Dans cette optique nos vidéos ont des caméras qui sont fixées le long des vidéos.

De plus, pour décider si une frame est "bonne" ou "mauvaise" nous utilisons un seuil sur l'accélaration qui est adaptatif à la vidéo. Il utilise la moyenne des aires pour choisir. Nous avons réglé à 10% par défaut ce seuil : ce qui veut dire que si l'accélaration sur les aires d'une frame augmente brutalement par rapport à cette moyenne, la frame est una anomalie. Procéder d'une telle manière est assez risqué, on pourrait imaginer des vidéos qui faussent intentionnellement la détection avec une moyenne petite mais avec une grande étendue sur les valeurs sur une longue vidéo pour lisser le tout. Symétriquement, une vidéo parfaite peut avoir une détection de mauvaises frames à cause des mouvements non constants.

Les aires ne sont donc pas très performantes pour déceler les problèmes de manière sûre, et sont en particuleir sensibles à la vitesse du déplacement, l'angle de caméra (en particulier si la caméra est proche ou loin), et de la proportion choisie pour le seuil.

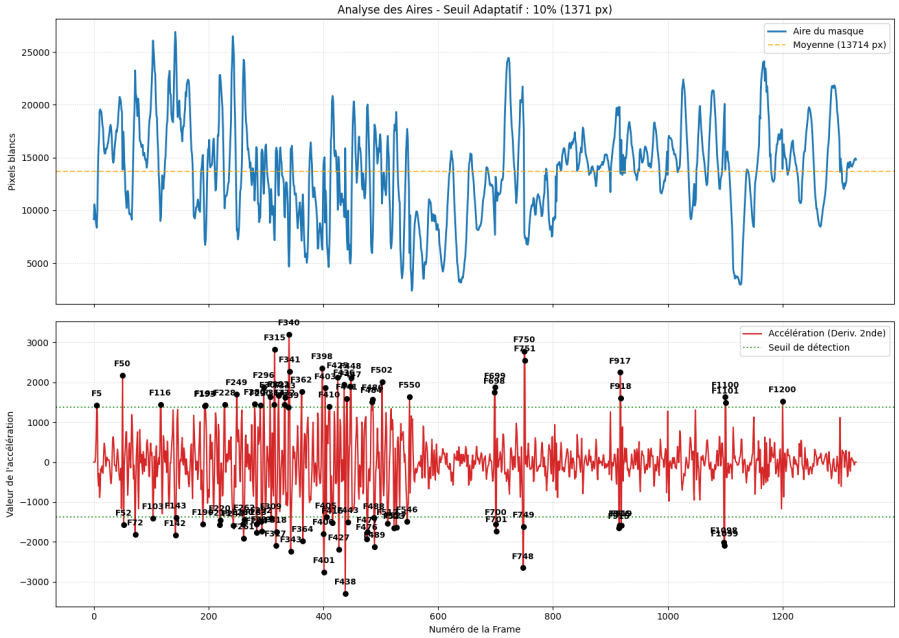

Figure 12 – Évolution temporelle des aires des masques et dérivée seconde associée.

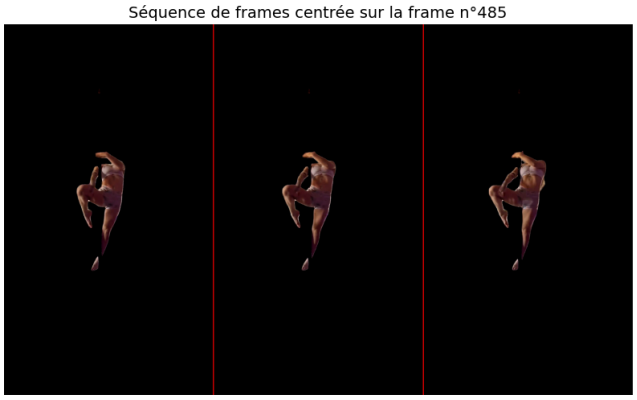

Figure 13 – Exemple de frame correcte mais où un bug est détecté sur les frames 484 (gauche) et 486 (droite), c'est un faux positif (la danceuse a la tête en arrière, il est normal qu'elle ne soit pas visible)

8.3 Distance de Bhattacharyya

Nous avons alors décidé de s'intéresser aux histogrammes des images, pour obtenir une représentation statistique des images, par conséquent différent de l'approche sur les aires.

Cette deuxième famille de métriques repose sur la comparaison d'histogrammes projetés des masques successifs. Nous avons associé ces histogrammes à la distance de Bhattacharyya (exprimée sous sa forme métrique de Hellinger) calculée à partir des projections suivant les axes horizontal et vertical. H représente les histogrammes des images et $\hat{H}$ les histogrammes normalisés.

$$H_{x,n}(i) = \sum_j I_n(i,j) \quad \text{et} \quad H_{y,n}(j) = \sum_i I_n(i,j) \tag{1}$$

$$\hat{H}_{x,n}(i) = \frac{H_{x,n}(i)}{\sum_i H_{x,n}(i)} \quad \text{et} \quad \hat{H}_{y,n}(j) = \frac{H_{y,n}(j)}{\sum_j H_{y,n}(j)} \tag{2}$$

$$Coef_{Bhattacharyya}(\hat{H}_n, \hat{H}_{n-1}) = \sum_k \sqrt{\hat{H}_n(k) \cdot \hat{H}_{n-1}(k)} \tag{3}$$

$$Distance_{Bhattacharyya}(\hat{H}_n, \hat{H}_{n-1}) = \sqrt{1 - BC(\hat{H}_n, \hat{H}_{n-1})} \tag{4}$$

Cette approche vise à mesurer la dissimilarité entre deux masques consécutifs ou entre un masque et une combinaison de ses voisins.

Il a été exploré deux méthodes, celle décrite avec les formules ci-dessus, et une autre en appliquant une fenêtre glissante autour de la frame étudiée (sans l'inclure elle-même). $H_{y,n}$ est à la place la pondération des images passées et futures autour de la frame d'histogramme $H_{x,n}$. Cela permet de lisser l'histogramme de comparaison et de faire ressortir d'avantage les frames particulièrement dégradées. Nous pensions que ce système serait à la fois plus robuste et plus performant, ce qui est discutable sur les deux points. On nomme cette méthode "Bhattacharyya avec overlay" dans la suite du rapport.

Appliqué à notre vidéo témoin, cette méthode sans overlay fournit une distance moyenne entre frames d'environ 0,177, soit un score de stabilité de 82,29 %. Avec overlay des masques voisins, la distance moyenne tombe à environ 0,156, ce qui correspond à un score de stabilité de 84,41 %. Ces résultats suggèrent que l'agrégation locale des masques voisins améliore légèrement la stabilité mesurée, ou d'un autre point de vue, que l'overlay n'augmente pas significativement le score mesuré.

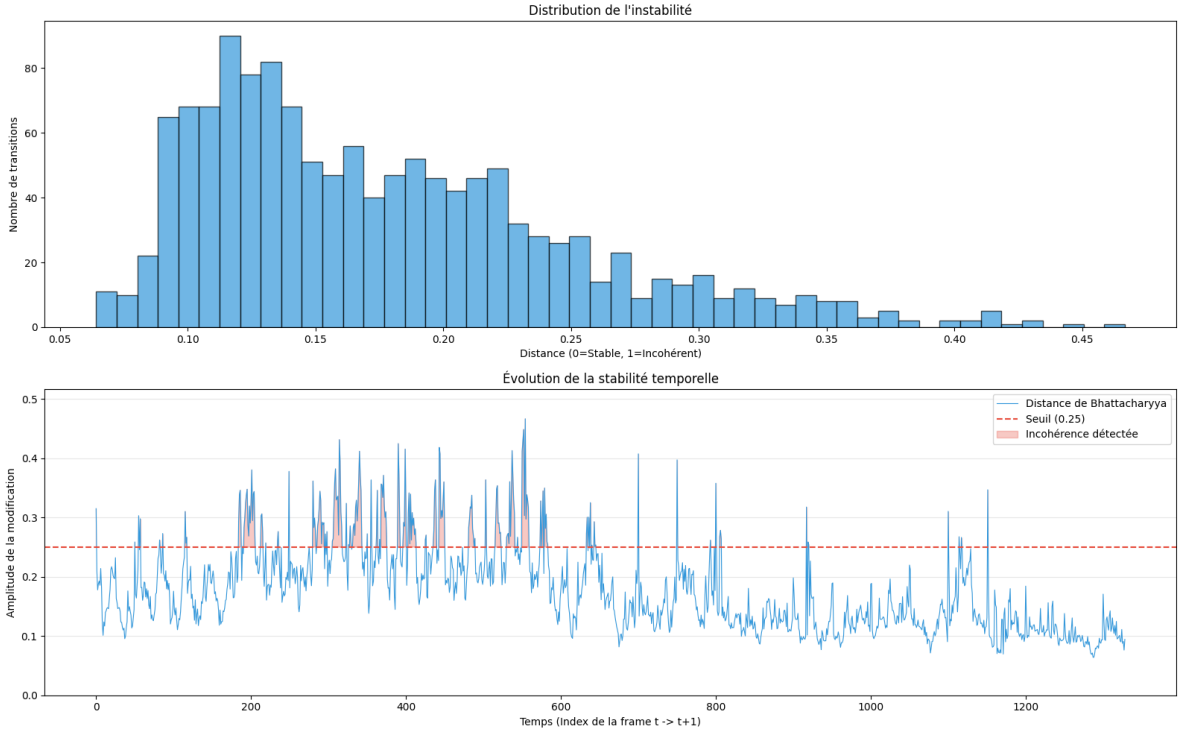

Figure 14 – Distribution et évolution temporelle de la distance de Bhattacharyya sans overlay.

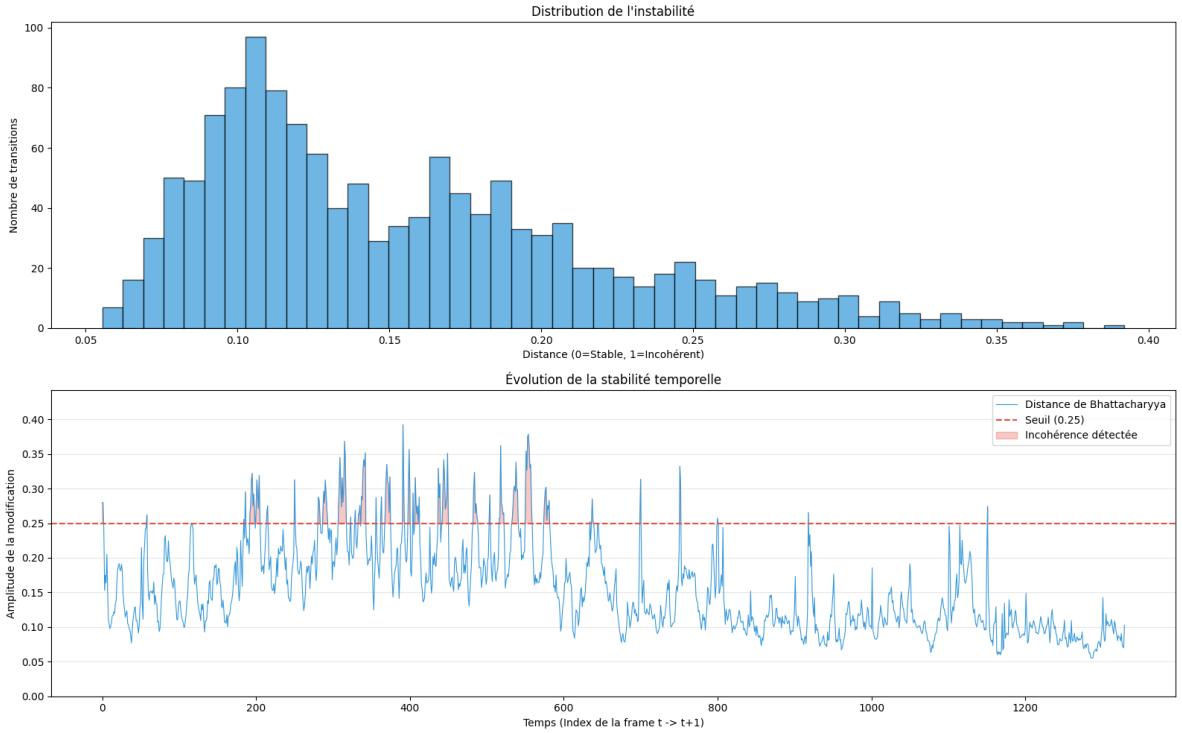

Figure 15 – Distribution et évolution temporelle de la distance de Bhattacharyya avec overlay de masques voisins.

8.4 Matrices de confusion

Des matrices de confusion ont été aussi générée sur notre vidéo témoin, en comparaison avec une annotation manuelle de l'ensemble des 1329 frames. Cette annotation a été réalisée de manière à punir mêmes les erreurs les moins visibles (parfois se jouant quelques vingtaines de pixels manquants). La donnée la plus importante est avant tout la capacité du modèle à repérer les faux positifs (bug effectif + bug repéré), puisque nous voulons idéalement trouver tous les vrai positifs. Cela nous permettrait par exemple de procéder à un second passage sur ces frames pour en ajuster la qualité.

Deux types d'évaluation sont à distinguer :

— une évaluation directe de l'état "BUG / OK" sur chaque frame correspondant à la différence avec l'annotation manuelle

— une évaluation des changements d'état, c'est-à-dire de la capacité de la métrique à détecter les transitions entre phases stables et phases dégradées (transitions Bug->OK et OK->bug)

Pour l'évaluation directe sans overlay, les résultats rapportés dans le notebook sont les suivants :

Classe	Précision	Rappel	F1-score
BUG (0)	0.38	0.15	0.22
OK (1)	0.62	0.85	0.72

Avec overlay, on obtient :

Classe	Précision	Rappel	F1-score
BUG (0)	0.38	0.11	0.17
OK (1)	0.62	0.90	0.73

Ces résultats montrent que la méthode détecte plus facilement les zones stables que les erreurs. Ce qui est normal car la majorité des frames sont bonnes. Autrement dit, elle est assez performante pour confirmer qu'une séquence est cohérente, mais plus limitée pour identifier de façon fiable toutes les frames réellement problématiques.

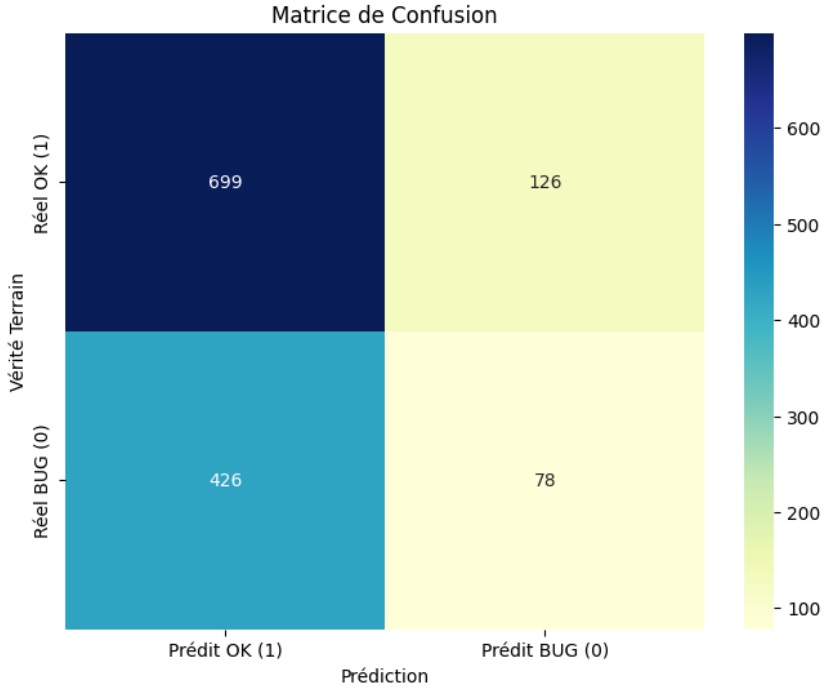

Figure 16 – Matrice de confusion associée à la détection des frames correctes et incorrectes, sans overlay.

Pour l'évaluation des changements d'état, l'indicateur semble naïvement performant puisque la distance de Bhattacharyya est grande lors des grandes variations des histogrammes.

Sans overlay :

Classe	Précision	Rappel	F1-score
STABLE (0)	0.94	0.86	0.90
CHANGEMENT (1)	0.13	0.28	0.17

Avec overlay :

Classe	Précision	Rappel	F1-score
STABLE (0)	0.93	0.90	0.92
CHANGEMENT (1)	0.10	0.15	0.12

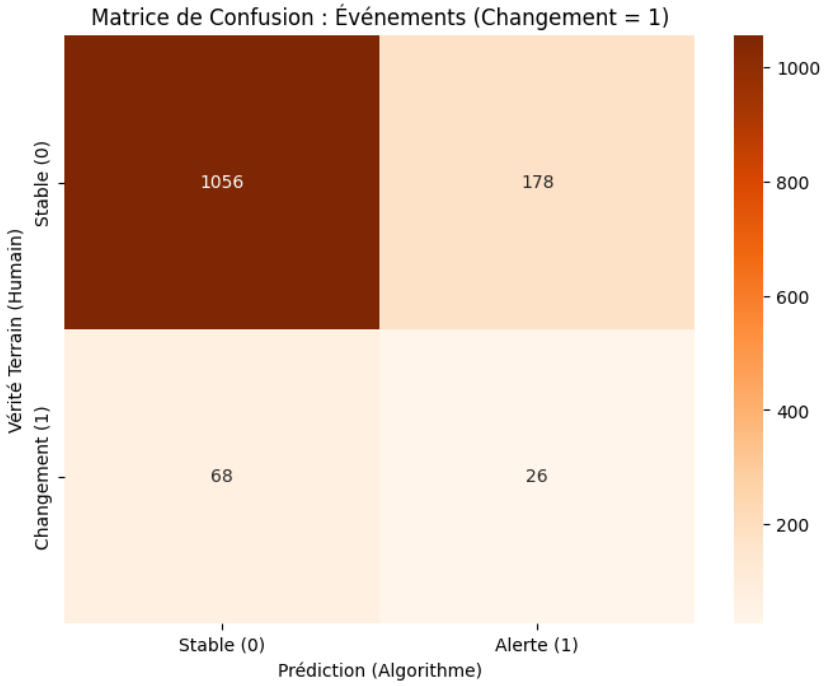

Figure 17 – Matrice de confusion associée à la détection des changements d'état, sans overlay.

On remarque alors que les frames de variations ne sont pas particulièrement bien détectées. On peut expliquer ça parce que les variations minimes ont été punies lors de l'annotation manuelle, qui sont donc dures à repérer depuis les histogrammes. On retrouve une asymétrie importante entre la détection des états stables et celle des changements, car les frames marquée BUG sont souvent adjacentes. Il y a encore moins de frames notées BUG dans ces matrices de corrélation pour cette raison.

La conclusion logique pour cette métrique est assez décalée par rapport à la réalité terrain et ne représente pas la bonne stabilité. Son score même si proche de la réalité, est davantage de l'ordre de la coincidence que de la performance de la segmentation.

8.5 Aidemos / Convert Body To 3D

Nous avons également exploré une nouvelle famille de métriques reposant sur l'outil `Aidemos / Convert Body To 3D`, correspondant au modèle `SAM 3D Body`. Contrairement aux indicateurs précédents, qui reposent directement sur le masque binaire ou sur ses projections, cette approche cherche à mesurer la cohérence géométrique du corps reconstruit à partir du calque vidéo. L'idée n'est donc plus seulement d'étudier la variation du contour, mais d'analyser la stabilité de la structure corporelle estimée entre deux frames consécutives.

L'intuition de départ est la suivante : si la segmentation reste cohérente au cours du temps, alors le corps reconstruit à partir des calques successifs doit lui aussi évoluer de manière régulière. À l'inverse, si une partie du corps disparaît brutalement, si le masque fusionne avec la barre, ou si le détourage devient instable, on s'attend à observer un saut plus important dans la reconstruction du squelette d'une frame à l'autre.

Concrètement, pour chaque frame retenue, nous reconstruisons d'abord un calque RGB recadré autour de la danseuse à partir de la frame originale et du masque binaire. Ce recadrage est ensuite fourni à `SAM 3D Body`, qui retourne notamment un ensemble de sommets 3D, des points-clés 3D et leurs projections 2D. La métrique est ensuite calculée non pas sur une frame isolée, mais sur des transitions consécutives $t \to t+1$ appartenant à un même run d'erreurs.

La grandeur principale retenue est une distance de Procrustes normalisée entre les points-clés 3D reconstruits sur deux frames successives. Si l'on note $K_t = (p_{t,1}, \ldots, p_{t,m})$ et $K_{t+1} = (p_{t+1,1}, \ldots, p_{t+1,m})$ les ensembles de points-clés 3D valides aux instants t et $t+1$, on cherche à comparer les deux configurations après recentrage et normalisation d'échelle. Le score de variation squelettique 3D peut alors s'écrire sous la forme :

$$d_{3D}(t, t+1) = \min_{R,s} \frac{1}{m} \sum_{j=1}^{m} \|\tilde{p}_{t,j} - sR\tilde{p}_{t+1,j}\|\,, \tag{5}$$

où R désigne une rotation orthogonale, s un facteur d'échelle, et $\tilde{p}_{t,j}$ les points-clés centrés. Cette distance vise à mesurer un changement de structure ou de posture indépendamment des simples effets de translation ou de changement de taille apparente.

En complément, une distance normalisée en 2D a également été calculée sur les projections des points-clés. Cette quantité n'est pas utilisée comme score principal, mais comme indicateur auxiliaire permettant de vérifier si les variations observées en 3D se retrouvent aussi dans la géométrie projetée de l'image.

Nous avons étudié deux versions de cette métrique. Une première version, dite « rapide », applique directement `SAM 3D Body` aux runs d'erreurs à analyser, puis calcule les sauts 3D et 2D sur les transitions consécutives de ces runs. Une seconde version, plus longue, ajoute une analyse complémentaire en comparant les transitions issues des runs d'erreurs à un petit ensemble témoin de transitions choisies hors de ces runs. Cette deuxième version est plus coûteuse, mais permet de mieux situer les scores obtenus par rapport à des zones supposées stables de la vidéo.

Sur la vidéo témoin, l'exécution longue appliquée aux transitions jugées anormales par la métrique précédente met en évidence huit runs courts : $(349, 350)$, $(424, 425)$, $(438, 439)$, $(549, 550)$, $(699, 700)$, $(749, 750)$, $(799, 800)$ et $(1099, 1100)$. Parmi eux, cinq transitions ont produit un score 3D exploitable, tandis que trois cas n'ont pas permis de reconstruire de squelette complet. Les scores 3D valides observés sont compris approximativement entre 0,21 et 0,81, les valeurs les plus élevées apparaissant notamment autour des transitions $424 \to 425$ et $549 \to 550$.

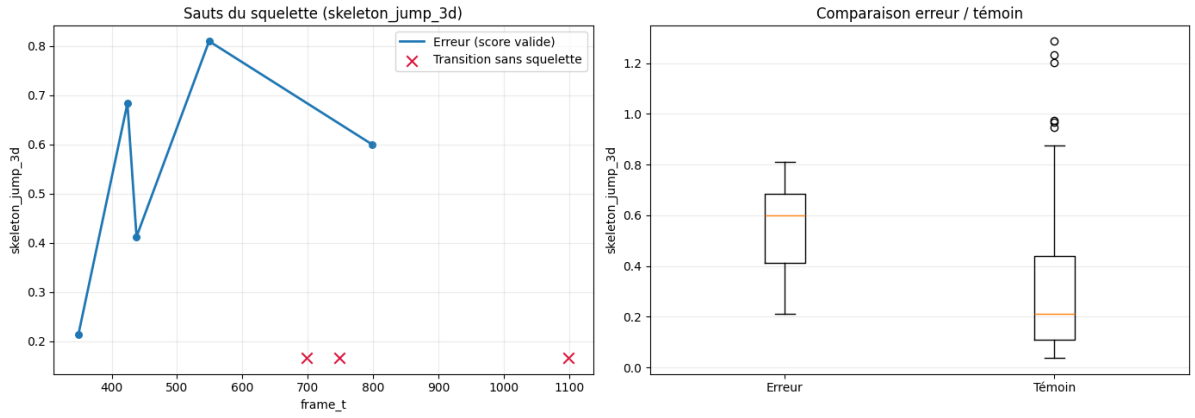

Figure 18 – Évolution des sauts du squelette estimés par `SAM 3D Body` sur les transitions anormales détectées (gauche) et comparaison entre transitions d'erreur et transitions témoins (droite). Les croix rouges correspondent aux transitions pour lesquelles aucun squelette valide n'a pu être reconstruit.

La figure 18 montre d'une part que certaines transitions se distinguent nettement des autres par un saut 3D important, et d'autre part que les transitions issues des runs d'erreurs présentent globalement des scores plus élevés que le groupe témoin. Sur ce cas expérimental, les scores d'erreur ont une moyenne d'environ 0,543, contre environ 0,307 sur le jeu témoin. La taille d'effet estimée est proche de 1, ce qui suggère une séparation non négligeable entre les deux groupes. En revanche, toutes les anomalies ne sont pas parfaitement séparées du témoin, ce qui montre que la métrique reste partiellement bruitée et ne permet pas à elle seule une discrimination complète.

Cette observation est cohérente avec la nature même de l'indicateur : il ne mesure pas directement la qualité du contour pixel à pixel, mais la cohérence du corps reconstruit. Il est donc particulièrement sensible aux erreurs qui modifient fortement la géométrie globale du sujet : disparition d'un membre, fusion partielle avec la barre, recadrage incohérent, ou changement brutal de silhouette. À l'inverse, il est moins adapté à la détection de défauts fins de segmentation, comme quelques mèches de cheveux manquantes ou une légère imprécision sur les bords.

La transition 549 → 550, qui correspond au plus grand saut 3D observé dans cette expérience, fournit un bon exemple qualitatif de ce comportement. La figure 19 montre les deux calques recadrés, ainsi que la reconstruction produite par `SAM 3D Body` sur chacune des deux frames.

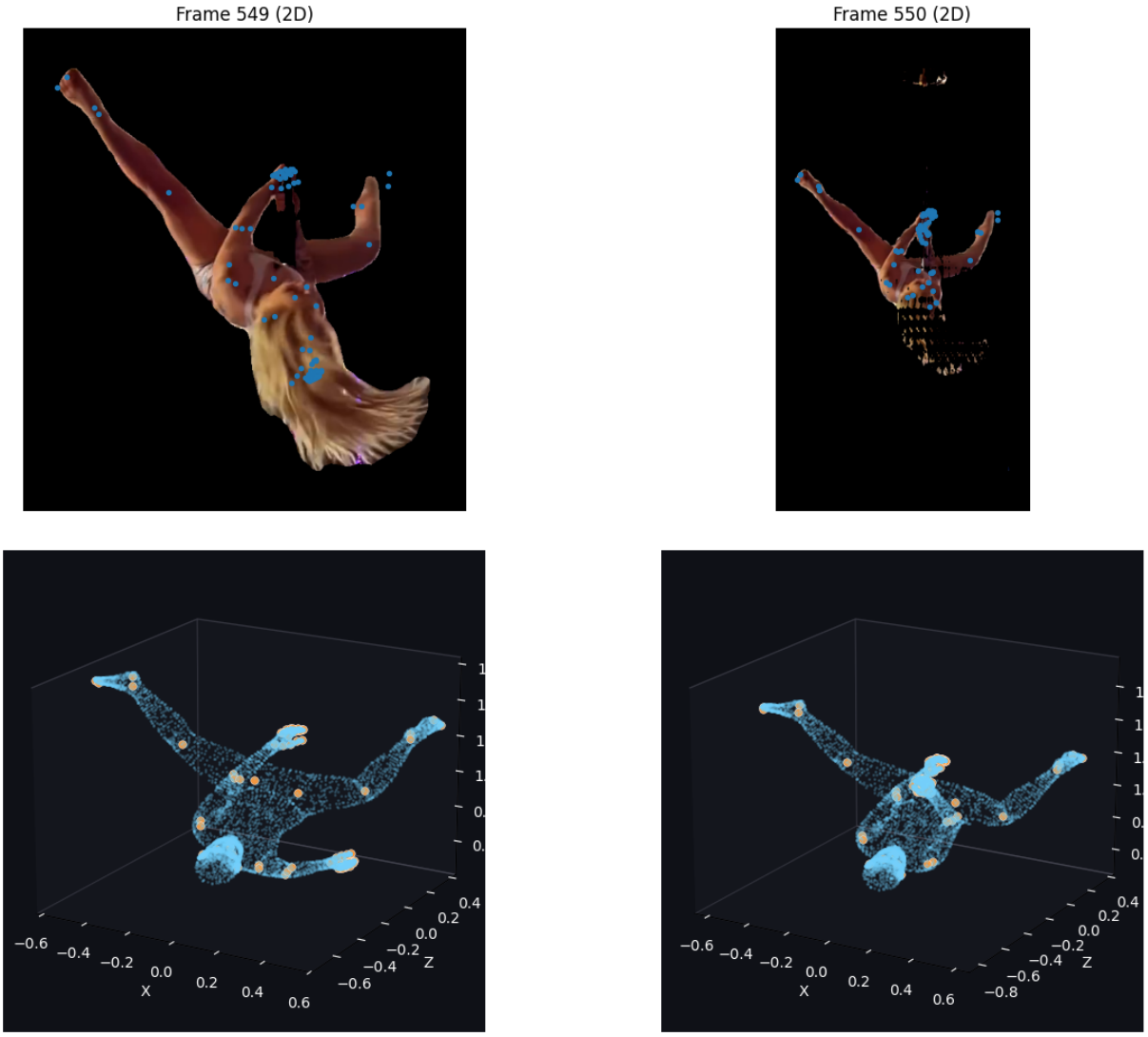

Figure 19 – Exemple de transition anormale pour la métrique `Aidemos / Convert Body To 3D` : comparaison des frames 549 et 550, avec visualisation des points-clés 2D et de la reconstruction 3D correspondante.

Cette seconde figure permet d'illustrer précisément et visuellement le type de situation que la métrique met en évidence (ici le bras droit de la pole danceuse qui change totalement de position de la frame 549 à 550 dû à un mauvais calque à la frame 549). Elle permet de relier le score numérique à une variation effective de la reconstruction corporelle entre deux frames successives.

Cette approche présente cependant plusieurs limites. La première est son coût computationnel élevé : le modèle est appliqué image par image, ce qui rend l'analyse complète nettement plus lente que les métriques précédentes. La seconde est qu'il s'agit d'un outil de reconstruction monoculaire sur image

unique, détourné ici pour une analyse vidéo ; la stabilité temporelle n'est donc pas imposée par le modèle lui-même, mais reconstruite a posteriori par comparaison de frames successives. Enfin, certains cas difficiles conduisent à l'absence totale de squelette reconstruit, ce qui complique l'interprétation des scores et oblige à distinguer explicitement les transitions réellement mesurées des échecs d'inférence.

En résumé, la métrique basée sur `Aidemos / Convert Body To 3D` apporte un point de vue complémentaire aux métriques fondées sur l'aire ou sur les histogrammes. Elle ne remplace pas les indicateurs plus simples, mais elle permet de détecter des incohérences d'un autre type, davantage liées à la cohérence corporelle de la danseuse qu'à la seule forme du masque. Dans l'état actuel du projet, nous la considérons donc comme une métrique exploratoire prometteuse, particulièrement intéressante pour analyser des anomalies fortes, mais encore trop coûteuse et trop sensible à l'échec de reconstruction pour constituer à elle seule un score final de qualité vidéo.

9 Discussion et perspectives d'amélioration

Le travail réalisé met en évidence plusieurs enseignements.

Tout d'abord, l'analyse de pose et la segmentation répondent à des besoins différents mais complémentaires. L'analyse de pose permet de structurer la scène et de localiser approximativement la danseuse, tandis que la segmentation fournit l'information précise nécessaire à la production d'une vidéo détourée exploitable. Dans notre pipeline, la combinaison `NLF + SAM 2` constitue ainsi un compromis fonctionnel entre robustesse, disponibilité des outils et intégration pratique dans Colab.

Tout d'abord, l'analyse de pose et la segmentation répondent à des besoins différents mais complémentaires.

- Dans la `V3`, l'analyse de pose permet de structurer la scène et de localiser approximativement la danseuse, tandis que la segmentation fournit l'information précise nécessaire à la production d'une vidéo détourée exploitable. La combinaison `NLF + SAM 2` constitue ainsi un compromis fonctionnel entre robustesse, disponibilité des outils et intégration pratique dans Colab.
- La `V4` modifie cependant cet équilibre. En remplaçant totalement `NLF` et `SAM 2` par `SAM 3`, elle supprime l'étape de pose comme guide de segmentation et produit des calques plus cohérents sur la vidéo témoin, notamment sur les parties fines. En revanche, cette amélioration qualitative s'accompagne d'un coût computationnel beaucoup plus élevé. Les deux approches apparaissent alors comme complémentaires : la `V3` reste plus légère et mieux adaptée aux expérimentations rapides, tandis que la `V4` fournit une segmentation plus convaincante mais plus coûteuse.

Ensuite, la mise en place de métriques d'évaluation s'est révélée essentielle. Sans elles, l'analyse du pipeline resterait limitée à des impressions visuelles. Même imparfaites, les métriques basées sur l'aire, les histogrammes et les matrices de confusion permettent d'objectiver certaines faiblesses du système et d'identifier des frames à inspecter en priorité.

Enfin, le projet met aussi en évidence la difficulté d'obtenir une segmentation réellement robuste dans un contexte de danse. Les positions extrêmes, les auto-occlusions, la finesse des cheveux et la nécessité de conserver une cohérence temporelle rendent le problème plus difficile qu'une simple segmentation d'objet sur image fixe.

Les perspectives d'amélioration sont nombreuses. Parmi les plus importantes, on peut citer :

- L'optimisation de la `V4`, afin de bénéficier de la qualité de `SAM 3` avec un coût d'exécution plus raisonnable,
- L'enrichissement et le raffinement des métriques de qualité,
- L'automatisation plus poussée de la réinsertion sur un nouveau fond.

10 Conclusion

Au terme de ce TER, nous avons mis en place deux chaînes de traitement fonctionnelles permettant d'analyser des vidéos de pole danse :

— En estimant puis segmentant la position du danseur présent dans la scène,

— En le réinsérant sur un nouveau fond,

— Et en évaluant quantitativement la qualité de la segmentation vidéo obtenue.

Au terme de ce TER, nous avons mis en place deux chaînes de traitement exploitables pour la segmentation vidéo de pole dance :

— Le premier pipeline (`V3`), composé de `NLF + SAM 2`, constitue une démarche qualitative dans l'ensemble avec un temps d'exécution relativement court.

  — Cette approche est satisfaisante bien que les résultats ont des légers manques, que l'oeil humain remarque facilement (oscillations, pertes ponctuelles de membres), ce pipeline est alors difficilement utilisable dans le contexte de deepfakes, mais présente tout de même une bonne stabilité pour obtenir un aperçu en peu de temps.

  — La `V3` est donc particulièrement adapté aux expérimentations rapides et à l'étude des métriques d'estimation de qualité de reconstruction vidéo et reste donc une bonne base méthodologique pour une exploration rapide sur plusieurs vidéos.

— Le second pipeline (`V4`), composé de `SAM 3`, produit un rendu globalement bien plus qualitatif que la `V3`, au prix d'une exécution plus longue.

  — Cette approche rend des résultats très convaincants, parfois proches de la perfection, pouvant tromper un oeil non-averti, ce qui la rend adaptée dans le contexte des deepfakes. En contrepartie, le coût temporel d'exécution est beaucoup plus élevé (environ 4 à 5 fois plus long que la `V3`).

  — La `V4` apparaît donc comme une alternative de meilleure qualité visuelle pour la segmentation elle-même.

Les deux approches doivent à ce stade être vues comme complémentaires plutôt que concurrentes : la `V3` est plus économique et plus flexible, tandis que la `V4` plus coûteuse mais plus performante qualitativement.