AirDrums : Jouer De La Batterie Avec La Webcam De Son Ordinateur

Karim Khaldi : karim.khaldi@student-cs.fr
Nicolas Noblot : nicolas.noblot@student-cs.fr

16 avril 2023

Résumé

Jouer de la batterie est un loisir très divertissant. Qui n'a jamais rêvé de jouer des percussions dans un groupe de rock ? Cependant, ce loisir peut être difficile d'accès de part le coût du matériel et son encombrement. Dans cette optique, nous avons implémenté un logiciel pour jouer de la batterie sur la webcam d'un ordinateur. Ce programme se veut être accessible à tous et ludique. Il repose essentiellement sur la détection des mains de l'utilisateur. Pour réaliser cette tâche, nous étudions différentes approches dans cet article.

Mots-clés : Vision par ordinateur, batterie, caméra, détection des mouvements des mains

1 Introduction

Dans cet article, nous présentons algorithme de vision par ordinateur pour jouer de la batterie en temps réel avec une caméra d'un ordinateur. L'algorithme utilise une caméra pour détecter les mouvements des mains de l'utilisateur et produire des sons de batterie en fonction de leur position sur l'écran. Nous avons mis en place une interface utilisateur qui permet aux utilisateurs de personnaliser les sons de batterie.

Cet algorithme offre des avantages significatifs par rapport aux méthodes traditionnelles de jeu de batterie. Les contraintes actuelles pour jouer de la batterie incluent le besoin d'espace pour installer les différents instruments, le coût élevé de l'équipement et la difficulté de transporter la batterie pour des performances en direct. Notre algorithme permet aux joueurs de batterie de s'affranchir de ces contraintes. De plus, il offre de potentielles applications pratiques dans différents contextes notamment l'improvisation dans des performances live et l'éducation musicale.

La principale tâche délicate dans cet algorithme consiste à détecter et traquer les mains de l'utilisateur. Nous avons implémenté différentes solutions pour réaliser la tâche et les comparons dans cet article.

2 État de l'art

La détection de la main dans une image est une tâche importante dans de nombreuses applications, notamment la reconnaissance de gestes, la réalité augmentée, la robotique, la sécurité et la surveillance. Les approches de détection de la main peuvent être divisées en deux catégories principales : des méthodes de vision par ordinateur et les méthodes d'apprentissage.

Parmi les méthodes classiques de vision ordinateur, on retrouve des procédures basées sur des règles utilisent des heuristiques et des connaissances préalables pour détecter la main dans une image. Ces méthodes sont souvent basées sur des caractéristiques telles que la couleur, la forme ou la texture de la main [8]. L'article de *Toni et al.* [9] a recours à différentes méthodes basées sur de la segmentation par seuillage à partir d'histogrammes et du clustering combinés de la détection par enveloppe convexe pour détecter la main. *Samiappan et al.* [3] utilisent un seuillage dans l'espace de couleurs HSV avec des méthodes de morphologie mathématiques pour segmenter la main. *Rios Sioria et al.* [2] ont aussi recours à cette technique de seuillage dans l'espace de couleurs HSV combinée avec une estimation d'enveloppe convexe pour réaliser une estimation de poses de la main dans une image.

Cependant, ces approches ont souvent des limitations en termes de précision et de robustesse, en particulier en présence de changements d'éclairage, de flou ou de mouvement rapide de la main [9], [3], [2]. D'autres techniques ont été développées reprenant les idées des méthodes classiques mélangées avec de l'apprentissage automatique. Ainsi, l'article *Mao et al.*fait appel au filtrage de couleur mêlé à des techniques de boosting pour segmenter la main en temps réel dans des environnements complexes. L'utilisation de l'apprentissage automatique permet de rendre la détection plus robuste aux changements d'arrière-plan [6].

L'avènement du deep learning depuis une dizaine d'année et en particulier le développement des réseaux de neurones convolutifs CNN ont apporté de nouvelles solutions pour la détection de la main dans des images. De ce fait, l'article de [4, Gao et al.] propose un architecture de réseaux de neurones appelée FF-SSD basée sur des couches Resnet et VCG16 pour réaliser la détection de mains en temps réel sur des images. D'autre part, *Oberweger et al.* [7] proposent leurs propres architectures de réseaux convolutifs pour détecter la main et les joints des articulations afin d'estimer la pose 3D du membre du corps.

3 Méthodologie

3.1 Méthodologie générale

La figure 1 expose les étapes successives qui permettent de déclencher un son à partir des images d'une caméra d'un ordinateur.

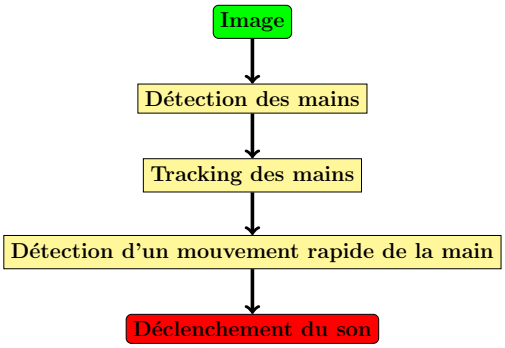

FIGURE 1 – Les différentes étapes de l'algorithme AirDrums

La première étape consiste à extraire une image de la caméra. Ensuite, un algorithme de détection se charge de localiser les mains dans l'image. Nous proposons plusieurs solutions pour cette tâche qui sont décrites plus loin dans la section 3.2.
Une fois les mains détectées, un simple algorithme de tracking enregistre la position des mains à savoir un rectangle encadrant la main. D'une image à la suivante, l'algorithme de tracking compare les positions des centres des rectangles trouvés sur la nouvelle image avec celles détectées sur l'ancienne image. Si les centres de deux rectangles sont suffisamment proches d'une image à la suivante, l'algorithme considère qu'il s'agit du même objet et met à jour sa position. Si la trace d'un objet est perdue sur 5 images consécutives, l'algorithme arrête de suivre l'objet.

Pour chaque objet enregistré par l'algorithme de tracking, on calcule la distance du centre de l'objet aux sommets du rectangle encadrant l'objet. Cette distance est mise à jour à chaque image de la vidéo. De plus, chaque objet garde en mémoire la variation de cette distance entre deux images successives. Un seuillage est effectué sur cette variation. Si la variation est positive et supérieure au seuil, un son est déclenché. Cette variation permet d'estimer le niveau de zoom de l'objet. Nous avons souhaité déclencher un son lorsque l'utilisateur déplace sa main rapidement de l'arrière vers l'avant (soit de l'arrière-plan vers l'écran).

Différents sons peuvent être joués durant l'utilisation. Le son joué dépend de l'endroit où est détecté le mouvement rapide de la main. Comme le montre la figure 3, 6 zones différentes ont été délimités sur l'écran de l'utilisateur avec sur chacune, le nom du son déclenché quand un mouvement rapide de la main est détecté dans cet zone. Les sons produits appartiennent à différents équipements de batterie ce qui donne son nom à l'algorithme AirDrums. Ils ont été pris de la banque de sons publique freewavessamples `https://freewavesamples.com/sample-type/drums`. Les sons sont produits à l'aide du module *simpleaudio* de Python.

3.2 différentes méthodes de détection de la main utilisées

Détection par couleur

La première méthode de détection de la main consiste à détecter les pixels de l'image qui correspondent à la couleur de la peau de l'utilisateur. Pour ce faire, une analyse des niveaux de couleurs de l'image à chaque instant est effectuée, et un filtre est appliqué pour ne garder que les pixels correspondant à la couleur de la peau. Dans un second temps, un noyau morphologique est appliqué afin de débruiter l'image. Celui-ci consiste en l'utilisation d'un noyau gaussien et d'une succession de dilatations, érosions sur une surface choisie comme elliptique. Enfin, un masque est appliqué au centre de l'image afin de retirer le visage de l'utilisateur et ne pas confondre l'algorithme. Suite à celà, un algorithme est ensuite appliqué afin de détecter le contour de la main et de créer un bloc rectangulaire aux contours de celle-ci.

Détection par soustraction d'arrière-plan

La seconde méthode de détection se base sur l'utilisation d'un algorithme de soustraction d'arrière-plan. Cet algorithme est basé sur une soustraction de l'image courante avec l'image précédente afin de déterminer les pixels qui ont changé. Une fois ces pixels détectés, on cherche alors les contours de la main via des seuils de couleurs, suivis d'une approximation polynomiale dudit contour, et enfin d'un rectangle entourant le seuil obtenu afin de définir la main.

Détection par machine learning

La méthode de détection via un algorithme de deepLearning effectuée à titre comparatif se base essentiellement sur l'utilisation d'un réseau de neurones pré-entraîné sur le dataset COCO [**?**] qui permet de détecter les objets présents dans une image. Le réseau de neurones utilisé est un modèle DeepLabV3 [1] avec un ResNet50 [5] pré-entraîné sur le dataset COCO. Celui-ci ne sert à l'origine qu'à de la détection d'objets. Nous avons donc effectué un fine-tuning de ce modèle afin de détecter les mains de l'utilisateur.

FIGURE 2 – Exemple d'image du dataset EgoHands retraitée par l'algorithme.

Les données d'entraînement utilisées sont basées sur le dataset publique EgoHands qui contient des images de mains annotées, ainsi qu'une quarantaine de vidéos de celles-ci. Notre

modèle est finetuné de sorte à retourner un filtre sur la zone où les mains sont détectées, après quoi un rectangle est réalisé afin d'entourer l'objet trouvé. Un checkpoint afin d'éviter de ré-entrainer celui-ci est disponible sur notre google drive

Création sonore

Une fois la main détectée, la création du son est réalisée en observant une variation de la position du rectangle ayant détecté la main.

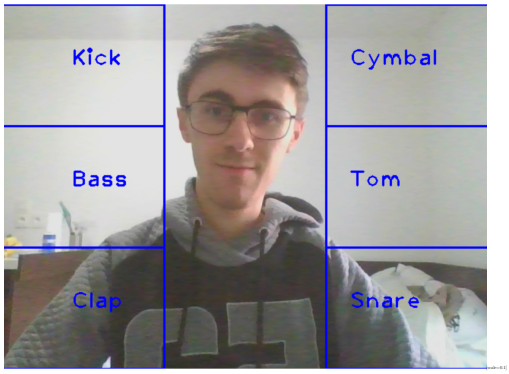

FIGURE 3 – Interface de l'algorithme

L'interface de l'algorithme est la suivante. Elle permet de choisir la méthode de détection de la main, ainsi que le son à jouer en déplaçant sa main de façon appropriée au bon endroit de l'écran.

Celui-ci se base sur la détection des déplacements du centre du réctangle créé lors de l'étape précédente. Lorsque ce centre approche d'un instrument, celui-ci est enclenché.

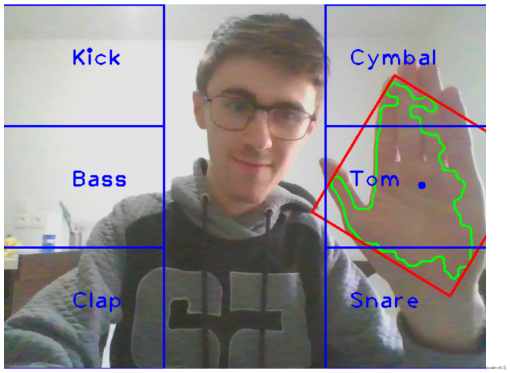

FIGURE 4 – Un exemple de détection de la main par l'algorithme

4 Résultats

La détection de main par couleur est la méthode la plus rapide. Elle est aussi la plus simple à mettre en place. Cependant, elle ne fonctionne que si la main est bien éclairée et si elle est de couleur claire. La détection par mouvements de l'utilisateur est nettement plus précise. Elle fonctionne même si la main est de couleur foncée, celle-ci se basant que sur un changement successif de l'image. La détection par machine learning est la méthode la plus lente. Elle est aussi la plus précise, fonctionnant sur différents type d'éclairages, avec différentes couleurs de peau. Elle est la plus compliquée à mettre en place, cependant, elle fonctionne même si la main est mal éclairée et si elle est de couleur foncée. Notons notamment l'absence de continuité dans les contours de mains détectées par l'algorithme, ainsi une main peut être détectée par morceaux et nécéssiter un re-traitement de celle-ci afin d'ob-

tenir une détection plus complète.

5 Discussion

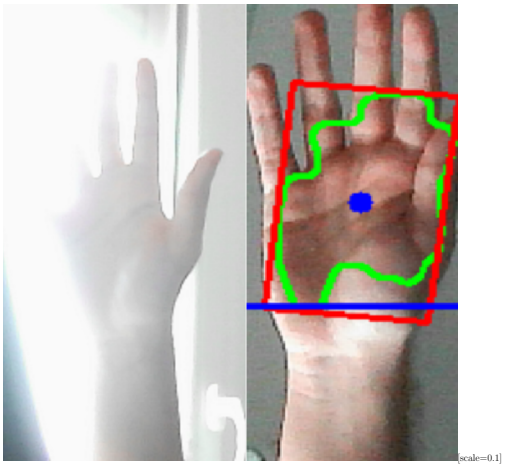

FIGURE 5 – Un exemple de main non détectée/détectée en fonction de l'éclairage

La méthode de deeplearning est la plus précise, mais aussi la plus lente, et la moins contrôlée. Les limites des méthodes de détections de features utilisées est qu'elles ne fonctionnent que si la main est bien éclairée et si elle est de couleur claire, et donc une suite de ladite méthode aurait été d'implémenter des méthodes de détection de contours afin de gérer les cas où la luminosité freine la méthode précédente, ou d'appliquer des seuils de luminosité par zone et de normaliser les zones en question. Ce travail aurait probablement permis d'atteindre une performance similaire à celle de l'algorithme de deeplearning, mais avec une plus grande flexibilité.

6 Conclusion

Lors de ce projet, nous avons eu l'occasion de découvrir et d'utiliser différentes méthodes afin de tester de la détection de mains. Bien que nous n'ayons pas eu l'occasion de concilier les méthodes de deep learning avec des méthodes de détections de features usuelles, nous sommes parvenus à réaliser un système permettant de produire du son uniquement via de l'analyse visuelle, dans le but final de faire du airdrum, en mettant en oeuvre des techniques vues en cours et en comparant avec un algorithme de deep learning finetuné pour la tâche en question.

Références

[1] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv :1706.05587*, 2017.

[2] Sara E. Garza-Villarreal David J. Rios-Soria, Satu E. Schaeffer. Hand-gesture recognition using computer-vision techniques. 2013.

[3] Ashwin R.M Subbiah K.R.Suraj Krishna Dhanalakshmi.Samiappan, J.Jayesh. Implementation of morphological operations to recognise hand

gestures under complex backgrounds. *International Journal of Pure and Applied Mathematics*, 115(5), 2017.

[4] Qing Gao, Jinguo Liu, and Zhaojie Ju. Robust real-time hand detection and localization for space human–robot interaction based on deep learning. *Neurocomputing*, 390 :198–206, 2020.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[6] Gang-Zeng Mao, Yi-Leh Wu, Maw-Kae Hor, and Cheng-Yuan Tang. Real-time hand detection and tracking against complex background. pages 905 – 908, 10 2009.

[7] Markus Oberweger, Paul Wohlhart, and Vincent Lepetit. Hands deep in deep learning for hand pose estimation. *CoRR*, abs/1502.06807, 2015.

[8] Munir Oudah, Ali Abdulelah Al-Naji, and Javaan Chahl. Hand gesture recognition based on computer vision : A review of techniques. *Journal of Imaging*, 6 :73, 07 2020.

[9] Benussi Toni and Jurić Darko. A robust hand detection and tracking algorithm with application to natural user interface. In *2012 Proceedings of the 35th International Convention MIPRO*, pages 1768–1774, 2012.