Rapport de Projet
DriveGuardian IA

Assistant d'aide a la conduite base sur dashcam (Computer Vision / ADAS)

Auteur :	Marouane Zaouia
Établissement :	ENSAM Rabat
Filière / Spécialité :	Aéronautique & Automobile — Projet orienté IA / Computer Vision (ADAS)
Année :	2025
Lieu :	Rabat, Maroc

DriveGuardian IA - Assistant de sécurité routière

Détection de voie et proximité véhicule avec scoring de risque

Dans le cadre d'une candidature en cycle ingenieur

Note de lecture

Conformement aux consignes, **aucune valeur numerique, statistique de performance, ni resultat experimental n'est invente**. Les valeurs manquantes sont notees ***a completer apres execution***. Cette rigueur est volontaire : elle distingue le contenu *methodologique* (ce qui est certain : architecture, logique, choix techniques) du contenu *experimental* (ce qui depend de l'execution et des donnees).

Positionnement

DriveGuardian IAest un assistant d'aide a la conduite : il **aide a percevoir et anticiper**, mais ne remplace pas le conducteur.

Resume

DriveGuardian IAest un prototype d'aide a la conduite base sur l'analyse d'un flux dashcam. Le systeme fournit : (1) detection de voie (centre/proche/hors) et estimation d'offset lateral, (2) detection multi-vehicules (jusqu'a 3) avec pseudo-radar (gauche/centre/droite), (3) estimation heuristique de distance et zones qualitatives (safe/close/very_close), (4) detection approximative des clignotants des vehicules proches, (5) analyse de risque en temps reel (SAFE/WARNING/DANGER) avec score 0–100, (6) alertes audio intelligentes (warning.wav/danger.wav) avec anti-spam, priorite danger et options ON/OFF, (7) dashboard temps reel (OpenCV) avec texte de contexte, (8) post-analyse fin de trajet : export CSV, graphes PNG, rapport texte et fenetre de recommandations.

La V1 a ete testee sur une video YouTube d'un trajet reel a Los Angeles : https://www.youtube.com/watch?v=XEzXUjzXF3s&t=767s. Ce choix est pertinent pour une V1 (scenario realiste, reproductible, focus algorithmique). Une captation personnelle est une perspective V2.

Pourquoi ce projet est coherent academiquement ?

Au-dela d'une simple demo, DriveGuardian IAillustre une demarche d'ingenierie : transformation d'un probleme reel en cahier des charges, conception modulaire, gestion d'etats temps reel (audio/risque), et production de livrables (exports, figures, bilan).

Abstract (optional)

This report presents DriveGuardian IA, a dashcam-based driver-assistance prototype combining lane detection, multi-vehicle detection, heuristic distance estimation, real-time risk assessment with a 0–100 score, smart audio alerts, and end-of-trip analytics (CSV + plots).

Table des matières

Table des figures

Liste des tableaux

Liste des acronymes

ADAS Advanced Driver Assistance Systems

CV Computer Vision (Vision par ordinateur)

ROI Region of Interest (Region d'interet)

TTC Time To Collision

UI/UX User Interface / User eXperience

FPS Frames Per Second

1 Introduction

1.1 Securite routiere : charge cognitive et variabilite du monde reel

La conduite automobile est une tache continue de perception et de decision. Le conducteur doit interpreter des indices heterogenes : marquages au sol, positions relatives des vehicules, variations de vitesse, intentions implicites (ex. clignotants), et contraintes environnementales (lumiere, pluie, ombres, vibrations). Dans un contexte reel, les informations utiles peuvent etre degradees (marquages effaces), ambigues (ombre vs ligne), ou partielles (occlusion par un vehicule). Cette variabilite rend le developpement d'un assistant fiable et interpretable particulierement formateur sur le plan ingenierie.

1.2 Positionnement : aider sans surpromettre

Un aspect essentiel d'un projet ADAS est la **responsabilite de communication** : un prototype n'est pas un systeme homologue. DriveGuardian IAadopte volontairement une approche d'assistance : il **signale** des situations a risque et fournit un **bilan** de trajectoire, mais ne prend pas d'action. Cette posture est coherente avec une V1 basee sur des methodes classiques, et permet d'ameliorer progressivement robustesse, calibration et evaluation.

1.3 Motivation personnelle et efforts investis

Ce projet est ne d'une motivation tres concrète : transformer un simple flux video de dashcam en une lecture utile, interpretable et actionnable, a la maniere d'un assistant Advanced Driver Assistance Systems (ADAS) de premiere generation. Mon objectif n'etait pas de produire une « demo gadget », mais de construire un systeme complet, coherent, et presentable, qui met en valeur une progression d'ingenierie (architecture, robustesse, stabilite, interface, et post-analyse).

1.3.1 Pourquoi ce sujet ?

La securite routiere est un domaine ou l'IA et la vision par ordinateur ont un impact direct, car la scene de conduite est riche en indices visuels (voie, vehicules, intentions, distances relatives). Le choix d'une dashcam impose une contrainte volontaire : **camera unique, sans capteurs de distance**. Cette contrainte m'a oblige a raisonner en termes d'indicateurs interpretable (heuristiques, zones qualitatives, score de risque) plutot que de promettre une precision metrique non justifiee.

1.3.2 Efforts d'execution et demarche d'ingenierie

Le developpement de DriveGuardian IAm'a demande un travail continu d'iteration : tester, observer, corriger, stabiliser. Au-dela de l'algorithme, une part importante des efforts a porte sur la **robustesse** et l'**qualite de restitution** :

- resolution d'erreurs de ressources (fichiers introuvables, conventions de nommage, audio mp3 vs wav) ;
- conception d'une logique evenementielle pour l'audio (anti-spam, priorite danger, etats, cooldown) ;
- amelioration de la stabilite percue (lissage, persistance d'etats) pour eviter un assistant « instable » ;
- amelioration progressive du dashboard (hierarchie de l'information, texte explicatif, lisibilite) ;
- production de livrables fin de trajet (CSV, graphes PNG, bilan) pour passer d'une execution a une analyse exploitable.

Ce que ce projet demontre sur ma methode

J'ai volontairement traite DriveGuardian IAcomme un mini-produit technique : un pipeline temps reel, une interface lisible, des erreurs gerees proprement, et une post-analyse. Cette approche reflète ma motivation a aller au-dela du « code qui marche » pour viser un systeme utilisable et evolutif.

1.3.3 Progression : de l'image au systeme video

Ayant deja travaille sur un projet de reconnaissance faciale (plutot oriente image), ce projet marque une transition : **de l'analyse ponctuelle a un systeme temps reel**. Il m'a oblige a concevoir des mecanismes de stabilite, des etats, des priorites (audio), et une logique de fin de trajet (export et recommandations), ce qui correspond davantage a une realite de systemes embarques et d'IA appliquee.

1.4 Donnees de test (V1) : choix et pertinence

La V1 est testee sur une video YouTube (trajet reel a Los Angeles) : https://www.youtube.com/watch?v=XEzXUjzXF3s&t=767s. Ce choix est utile au demarrage pour trois raisons :

— **Realiste** : circulation, perspectives et interactions proches d'un usage reel.

— **Reproductible** : meme sequence pour comparer des versions successives.

— **Orientee ingenierie** : l'effort se concentre sur pipeline, stabilite et presentation.

En V2, une captation personnelle permettra d'introduire une calibration physique et des tests mieux controles.

FIGURE 1.1 – Exemple de scene utilisee pour la V1.

2 Cahier des charges

2.1 Objectifs fonctionnels (ce que le systeme doit faire)

Les objectifs sont structures autour de deux dimensions : **perception** (extraire des indices) et **assistance** (transformer ces indices en information utile pour l'utilisateur).

2.1.1 Perception de voie

Le module voie doit :

— extraire les lignes dominantes de la route dans la zone pertinente ;

— estimer un centre de voie et un offset lateral ;

— qualifier la situation de maniere interpretable (centre, proche ligne, hors voie).

Une attention particuliere est accordee a la stabilite temporelle : une detection instable degrade l'UX et l'utilite.

2.1.2 Perception vehicules et proximite

Le module vehicules doit :

— detecter jusqu'a trois vehicules pertinents (compromis entre richesse d'information et lisibilite) ;

— situer ces vehicules lateralement (gauche/centre/droite) ;

— fournir une estimation heuristique de proximite, sous forme de zones qualitatives.

La distance est volontairement presentee comme *heuristique* en V1, afin d'eviter une interpretation metrique non justifiee.

2.1.3 Analyse de risque et alertes

L'analyse de risque doit :

— fusionner les signaux voie et distance pour produire un niveau (SAFE/WARNING/-DANGER) ;

— proposer un score graduel (0–100) permettant une lecture fine et un tracage ;

— generer un texte court de contexte ("smart") reliant le risque a sa cause dominante ;

— declencher des alertes audio avec priorite danger et anti-spam.

2.1.4 Interface et post-analyse

Le dashboard doit etre **lisible, stable et hierarchise**. En fin de trajet, le systeme doit produire : CSV, figures PNG, texte de bilan et recommandations. Cette post-analyse transforme une demo en livrable exploitable.

2.2 Contraintes non fonctionnelles (qualite systeme)

2.2.1 Temps reel et latence

Le traitement doit suivre le flux video en limitant la latence. La mesure du Frames Per Second (FPS) est *a completer apres execution*, mais l'architecture est concue pour favoriser des operations localisees (ROI) et des calculs simples en V1.

2.2.2 Robustesse logicielle

Une part importante du projet concerne la robustesse :
— gestion de fichiers manquants (cars.xml, audio) ;
— compatibilite formats (mp3 vs wav) ;
— stabilite des affichages (Unicode/accents) ;
— exports propres sans fenetres parasites.

2.3 Indicateurs a completer apres execution

TABLE 2.1 – Indicateurs de validation (*a completer apres execution*).

Indicateur	**Valeur**
Duree analysee	**238.76 s**
Nombre de frames	**7164**
FPS moyen	**30 fps**
Repartition voie (centre/proche/hors)	**90.44% / 9.45% / 0.11%**
Nombre d'episodes WARNING / DANGER	**45 / 36**
Distribution vehicules detectes (0/1/2/3)	**0 : 1200 | 1 : 4000 | 2 : 800 | 3 : 164**
Stats distance heuristique (min/moy/max)	**14.46 / 38.29 / 63.16 m**
Frequence alertes audio warning/danger	**warning : 18 | danger : 7**

3 Architecture globale

3.1 Choix d'architecture : modularite et interpretabilite

L'architecture de DriveGuardian IAest volontairement modulaire : chaque module produit une sortie interpretable. Cette approche apporte deux benefices majeurs :

— **Debug** : on peut isoler une source d'erreur (voie, vehicules, audio).

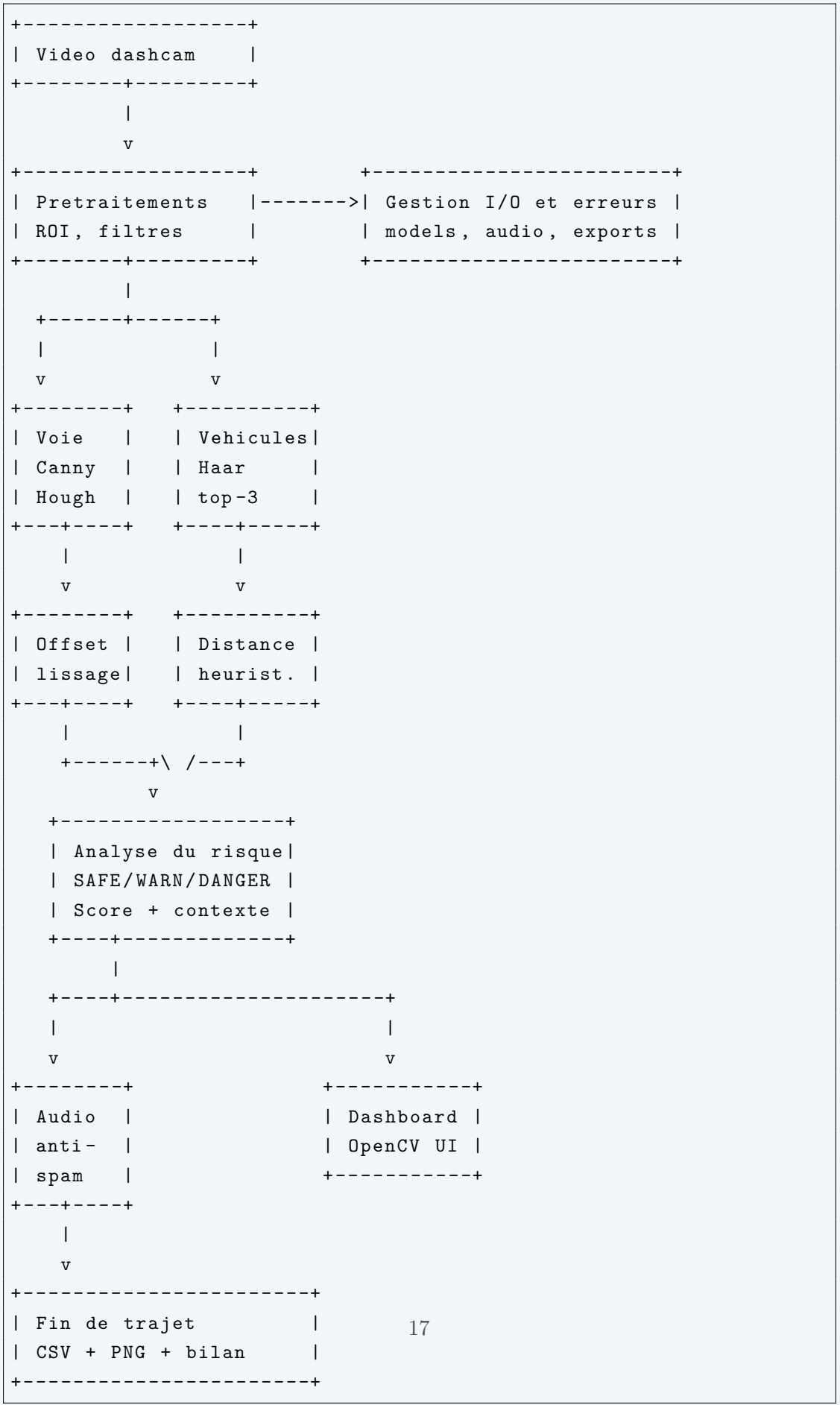
— **Evolution** : une V2 peut remplacer un module (ex. Haar -> YOLO) sans reecrire tout le systeme.

3.2 Pipeline global

Le flux de traitement suit la chaine : *frame -> perception -> fusion -> assistance -> export*. Une telle structure correspond aux logiques ADAS classiques : perception multi-capteurs (ici camera unique), decision interpretable (regles), et interaction (UI/alertes).

3.3 Schema ASCII du flux (compilable)

Pipeline global (ASCII)



3.4 Modes et configuration : city / highway

Le projet introduit deux modes car le contexte de conduite change les priorites :

— **City** : nombreuses interactions, stop-and-go, changements frequents. On cherche a limiter les fausses alertes et a privilegier la stabilite.

— **Highway** : vitesse plus constante, distances critiques. Les alertes distance/persistance prennent plus d'importance.

Les seuils et ponderations exactes restent *a completer apres execution.*

3.5 Mode demonstration : DEMO_MODE

Le `DEMO_MODE` permet une analyse courte (2–3 min) ou complete. L'objectif n'est pas d'optimiser artificiellement, mais de faciliter la demonstration et les tests rapides pendant le developpement.

4 Detection de voie

4.1 Approche V1 : traitement d'image interpretable

La detection de voie en V1 repose sur des techniques classiques, choisies pour leur transparence : l'utilisateur et le developpeur peuvent visualiser les contours, les segments et la reconstruction. Cette interpretabilite est importante pour un projet academique, car elle rend les limites evidentes et facilite une evolution vers une V2 deep learning.

4.1.1 ROI (Region of Interest)

La ROI est une decision d'ingenierie : elle restreint le calcul a la zone informative (partie basse), ce qui reduit bruit, faux positifs et cout de calcul. En pratique, la ROI est aussi un outil de stabilite.

4.1.2 Contours (Canny)

Le filtre Canny fournit une carte de contours. Son interet est de transformer une image complexe en indices geometriques. Les seuils de Canny sont *a completer apres execution*car ils dependent fortement de la luminosite et de la video.

4.1.3 Segments (Hough)

La transformee de Hough permet d'extraire des segments lineaires. Une etape cruciale consiste ensuite a trier/filtrer les segments (gauche vs droite, rejet des segments incoherents), puis a produire une ligne representative.

4.2 Offset et statut

Une fois un centre de voie estime, l'offset exprime la deviation laterale relative (en pixels ou unite derivee). Les seuils qui decident "centre/proche/hors" sont *a completer apres execution*. L'important est la logique : un statut est plus utile qu'une valeur brute lorsque l'utilisateur doit reagir vite.

4.3 Stabilisation temporelle

Sans lissage, le systeme peut alterner entre statuts a cause de bruit visuel. La stabilisation peut combiner :

— moyenne glissante sur l'offset ;
— maintien d'etat pendant un court intervalle si la detection se degrade ;
— hysteresis (seuils differents entree/sortie) pour eviter les oscillations.

Points cles (voie)

— La ROI et le lissage sont des leviers majeurs d'UX.
— La methode est interpretable, donc parfaite pour apprendre et expliquer.

Limites (voie)

— ombres, reflets, marquages effaces ou multiples ;
— scenes complexes (travaux, bretelles, intersections) ;
— absence de comprehension semantique (pas de segmentation).

FIGURE 4.1 – Contours Canny dans la ROI.

FIGURE 4.2 – Segments Hough et lignes finales .

5 Detection des vehicules

5.1 Choix V1 : cascade Haar (prototype rapide)

Le detecteur Haar est un choix pragmatique : il permet de construire un pipeline complet sans dependances lourdes. Dans une approche academique, il est acceptable tant que ses limites sont clairement identifiees et qu'il constitue une marche vers une V2 plus robuste.

5.2 Multi-vehicules et selection top-3

Afficher toutes les detections reduit la lisibilite. Limiter a 3 vehicules apporte :

— une interface plus claire ;
— moins d'instabilite visuelle ;
— un cout de calcul plus stable.

Le tri peut se faire par taille apparente, position ou stabilite temporelle (selon implementation).

5.3 Pseudo-radar (gauche/centre/droite)

Le pseudo-radar transforme la detection brute en information interpretable : ou se situe la menace potentielle. C'est un bon exemple d'ingenierie UI/UX : on ne se contente pas d'une bounding box, on construit une lecture rapide.

5.4 Distance heuristique et zones

Sans calibration, une distance metrique serait trompeuse. La V1 prefere une estimation heuristique (convertie en zones qualitatives). Ce choix protege l'utilisateur d'une interpretation erronnee et facilite l'analyse du risque (les regles manipulent des zones et non des metres).

Limites (vehicules/distance)

- faux positifs/faux negatifs selon angles et occlusions ;
- perspective et tailles vehicules variables ;
- estimation relative, non metrique.

6 Detection approximative des clignotants

6.1 Intention et utilite

Detecter un clignotant est une tentative d'anticipation : un vehicule lateral qui clignote peut annoncer un changement de file. Dans un assistant, meme un indice imparfait peut enrichir le contexte, a condition de le presenter comme "hint" (indice) et non certitude.

6.2 Approche V1 : indice temporel

Une detection approximative peut se baser sur une variation periodique de luminosite dans une zone plausible. L'idee centrale est temporelle : ce n'est pas une couleur unique qui definit le clignotant, mais une alternance relativement stable sur plusieurs frames.

Limites (clignotants)

- reflets, freinage, soleil couchant et compression video ;
- resolution insuffisante pour isoler une zone fiable ;
- a considerer comme fonctionnalite exploratoire.

FIGURE 6.1 – Affichage clignotant.

7 Analyse du risque (SAFE/WARNING/DANGER)

7.1 Pourquoi une analyse de risque ?

Dans un projet ADAS, la detection brute (voie/vehicule) n'est pas suffisante : il faut convertir l'information en action. L'analyse de risque agit comme un "chef d'orchestre" : elle choisit quoi afficher, quoi alerter, et comment expliquer l'etat a l'utilisateur.

7.2 Fusion de signaux et score interpretable

Le risque combine :

- la stabilite de la trajectoire (voie) ;
- la proximite (vehicules) ;
- la persistance (un risque bref n'a pas la meme signification qu'un risque durable) ;
- le contexte (vehicule lateral proche, indice de manoeuvre).

Le score 0–100 est un indicateur graduel (non metrique) utile pour la post-analyse et les graphes.

7.3 Formulation parametrique (sans valeurs)

Une forme generale peut etre :

$$score = \mathrm{clip}_{[0,100]}\Big(\sum_i w_i\, S_i\Big)$$

ou w_i et S_i sont *a completer apres execution*. L'important n'est pas la formule exacte, mais la logique : pondere des signaux interpretable et produire un etat stable.

7.4 Modes city/highway

- City : reduire les fausses alertes, mettre en avant la stabilite et les deviations persistantes.

— Highway : renforcer la sensibilite a la proximite et a la persistance, car les consequences d'une distance trop faible peuvent etre plus graves.

Les seuils restent *a completer apres execution.*

7.5 Texte explicatif "smart"

Un systeme efficace explique "pourquoi". Le texte de contexte est concu pour :

— identifier la cause dominante (voie vs distance) ;

— proposer une action simple (recentrage, augmenter l'ecart, vigilance).

Cette approche renforce la confiance et reduit l'effet "boite noire".

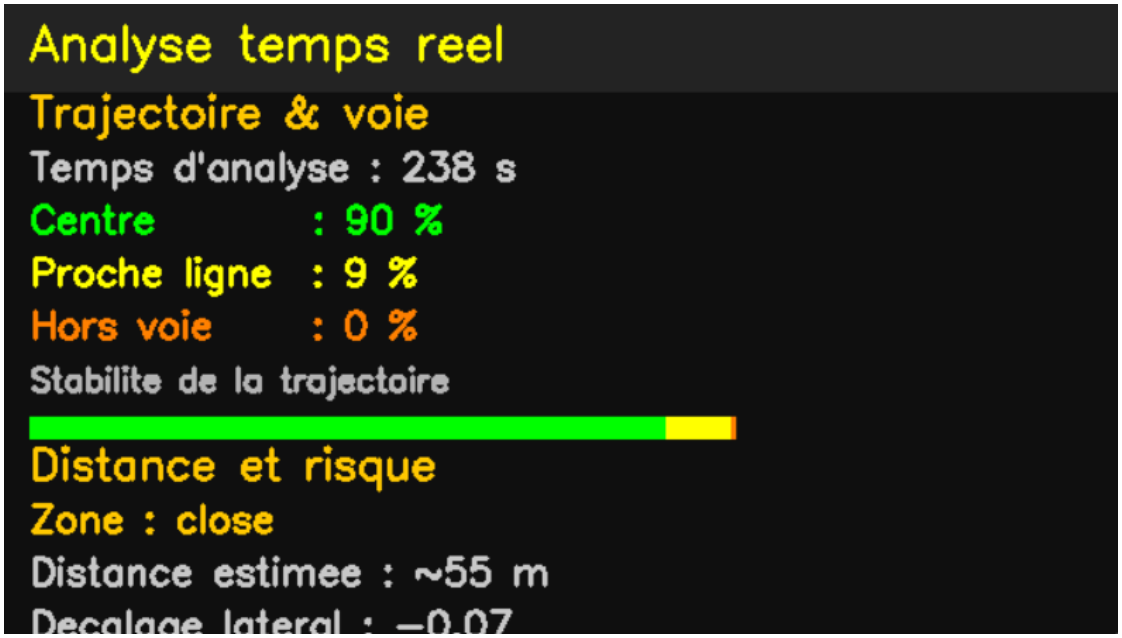


FIGURE 7.1 – SAFE/WARNING/DANGER + score + explication .

8 Alertes audio intelligentes

8.1 Objectif et contrainte humaine

L'audio est puissant mais intrusif. Un bon assistant doit alerter au bon moment sans fatiguer l'utilisateur. Le role du module audio est donc double :

— signaler un danger immediat ;

— eviter l'effet spam (alertes repetitives qui finissent ignorees).

8.2 Priorite danger et anti-spam

La priorite danger garantit qu'une alerte critique n'est pas masque par des warnings. L'anti-spam repose sur un cooldown et une gestion d'etat (derniere alerte + temps). Les valeurs temporelles sont *a completer apres execution*.

8.3 Robustesse : fichiers et formats

Le developpement a mis en evidence des problemes classiques en projet systeme :

— fichiers introuvables (chemins relatifs, conventions de nommage) ;

— incompatibilite mp3 vs wav ;

— lecture audio qui coupe si l'etat n'est pas gere.

Les corrections reposent sur des checks I/O et une logique d'etat coherente.

9 Dashboard temps reel (OpenCV)

9.1 UI/UX : hierarchie et lisibilite

Un dashboard utile met en avant l'essentiel :

— niveau de risque + justification courte ;

— statut voie + offset ;

— vehicules et zones de proximite ;

— details supplementaires uniquement si stable.

Ce choix limite la surcharge cognitive.

9.2 Unicode : accents affiches en " ? ?"

OpenCV peut afficher incorrectement certains caracteres accentues. Une solution V1 simple est de normaliser les caracteres avant affichage afin de garantir un rendu stable, meme si cela sacrifie certains accents.

Listing 9.1 – Normalisation Unicode (concept V1).

```
import unicodedata

def normalize_text(s: str) -> str:
    return ''.join(
        c for c in unicodedata.normalize('NFKD', s)
        if not unicodedata.combining(c)
    )
```

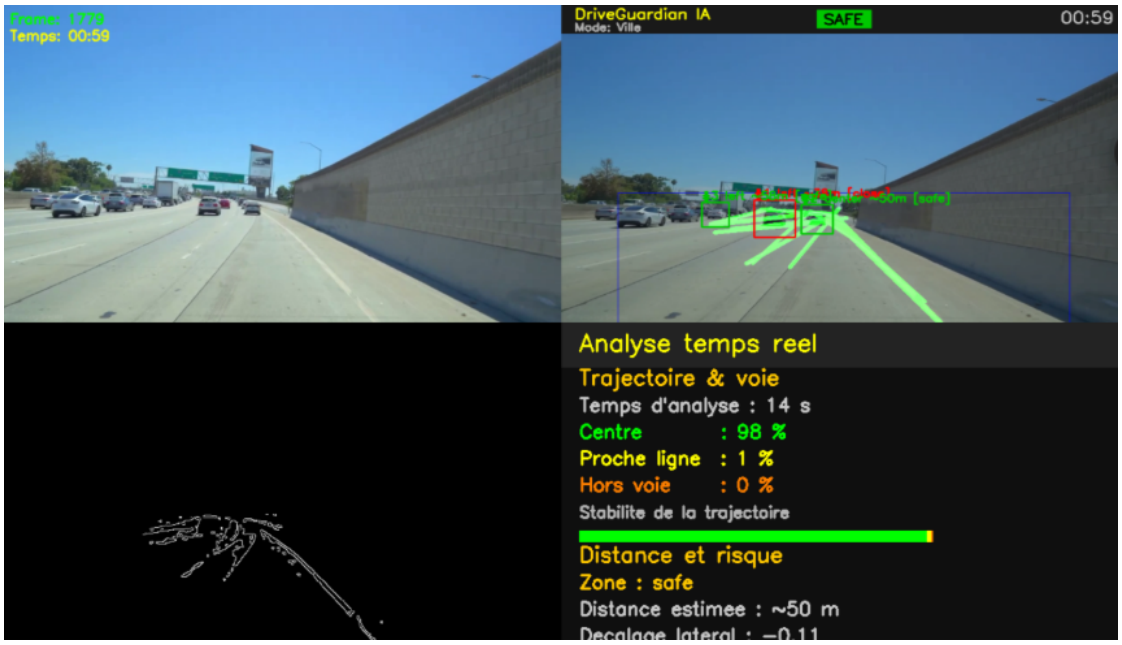


FIGURE 9.1 – Dashboard final OpenCV .

10 Export et post-analyse

10.1 Pourquoi exporter ?

Un projet academique gagne en valeur quand il produit des traces exploitables. L'export CSV transforme une session en donnees analysables : on peut tracer, comparer des versions, identifier des situations recurrentes et justifier des choix de seuils.

10.2 Graphes PNG

Les graphes (score vs temps, distance vs temps, distribution voie) rendent le bilan concret et presentable. Les figures sont generees en PNG pour etre directement integrables au rapport.

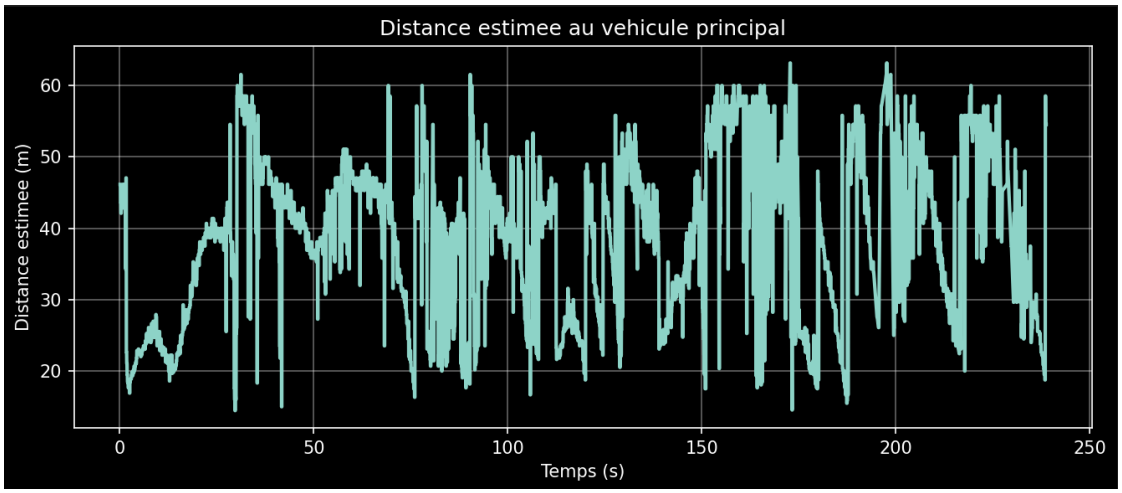


FIGURE 10.1 – Evolution de la proximite .

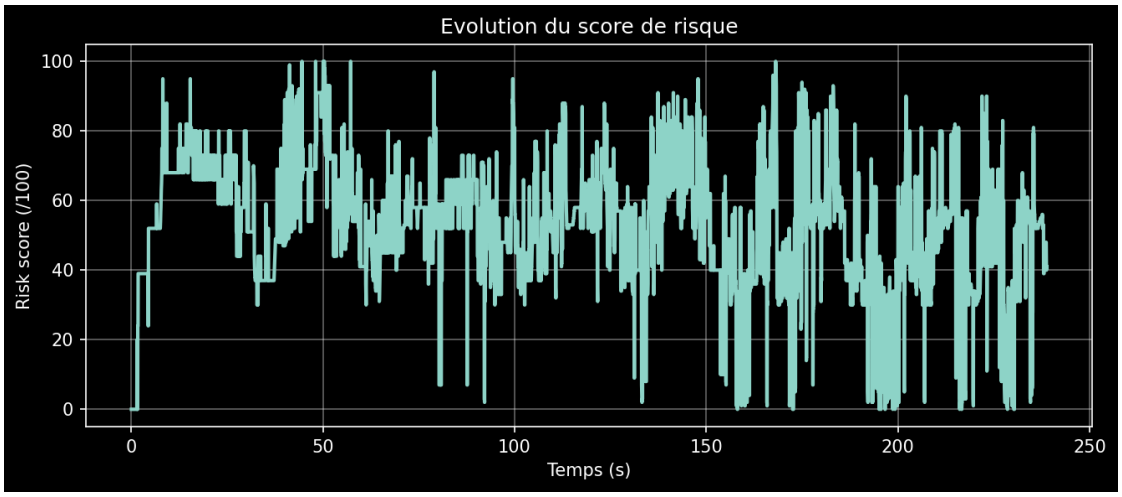


FIGURE 10.2 – Evolution du score de risque .

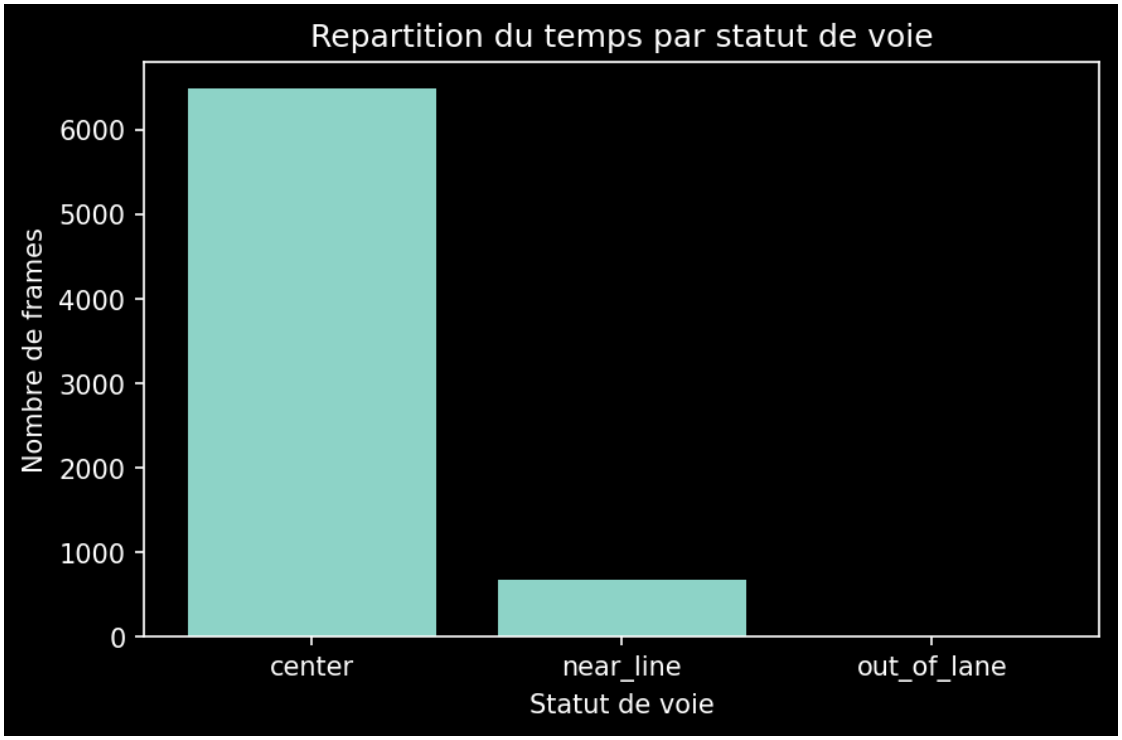


FIGURE 10.3 – Distribution des statuts de voie .

10.3 Bilan et recommandations

Le bilan final doit rester actionnable : identifier les contextes risqués et proposer des recommandations simples. Toutes les statistiques du bilan sont *a completer apres execution.*

11 Difficultes et apprentissages

11.1 Difficultes techniques (et pourquoi elles sont formatrices)

Les problemes rencontres sont typiques d'un systeme temps reel :

- erreurs I/O (fichiers manquants) : importance des checks et logs ;
- gestion audio : priorites et etats, au-dela du simple "jouer un son" ;
- stabilite UI : texte, encodage, lisibilite ;
- exports : produire des livrables propres et reproductibles.

Chaque probleme a pousse a une solution plus robuste, donc a une progression d'ingenierie.

Approche iterative

Tester -> observer -> corriger -> stabiliser -> embellir.

12 Ancrage dans mes etudes

12.1 Ce que mes etudes m'ont apporte (mise en pratique)

12.1.1 Programmation Python et conception systeme

DriveGuardian IAmobilise des competences transversales acquises et consolidees pendant mes etudes :

- structuration d'un programme en modules (voie, vehicules, risque, audio, export) ;
- gestion de flux (lecture frame par frame), gestion d'etats et evenements ;
- gestion de fichiers (chemins, erreurs, verification de ressources) ;
- production de livrables (CSV, figures, rapport) et reproductibilite.

Cette dimension « systeme » est centrale : en pratique, un assistant utile depend autant de la robustesse que de l'algorithme.

12.1.2 Raisonnement scientifique et validation

L'approche suivie est experimentale : formuler une hypothese (ex. lissage necessaire), observer un symptome (instabilite), proposer une solution (moyenne glissante / hysteresis), puis verifier sur la meme sequence video. Les indicateurs de performance sont notes *a completer apres execution*pour etre completes apres execution, dans un souci de rigueur.

Pourquoi ce chapitre est important pour un jury

Il montre que le projet n'est pas un assemblage de bibliotheques, mais une mise en pratique de competences d'ingenieur : raisonnement, modelisation, compromis, et qualite de restitution.

12.2 Formules et modeles mathematiques utilises (sans valeurs numeriques)

Cette section explicite les relations mathematiques sous-jacentes, sans inventer de parametres ni resultats.

12.2.1 Geometrie image : offset lateral de voie

Soit W la largeur de l'image. On note x_c la coordonnee horizontale du centre image et x_{voie} le centre de voie estime. L'offset lateral (relatif) peut s'ecrire :

$$x_c = \frac{W}{2}, \qquad \Delta x = x_{\text{voie}} - x_c$$

Une version normalisee (independante de la resolution) :

$$\Delta x_{\text{norm}} = \frac{\Delta x}{W}$$

La classification (centre / proche ligne / hors voie) repose sur des seuils τ_1, τ_2 :

$$\text{centre si } |\Delta x_{\text{norm}}| \leq \tau_1, \qquad \text{proche si } \tau_1 < |\Delta x_{\text{norm}}| \leq \tau_2, \qquad \text{hors voie si } |\Delta x_{\text{norm}}| > \tau_2$$

avec τ_1, τ_2 ***a completer apres execution***.

12.2.2 Transformee de Hough : representation de droites

Une droite detectee par Hough peut s'exprimer sous forme normale :

$$\rho = x \cos\theta + y \sin\theta$$

Cette representation est robuste pour agregation de segments. Une fois une droite retenue, on peut repasser en forme affine :

$$y = ax + b$$

Les parametres a, b servent ensuite a extrapoler et a estimer une trajectoire de voie dans la ROI.

12.2.3 Filtrage temporel : moyenne glissante et EMA

Pour stabiliser un signal s_t (offset, score, distance heuristique), une moyenne glissante sur N points :

$$\overline{s}_t = \frac{1}{N} \sum_{k=0}^{N-1} s_{t-k}$$

Une alternative plus reactive est la moyenne exponentielle (EMA) :

$$\widehat{s}_t = \alpha s_t + (1-\alpha)\widehat{s}_{t-1}$$

ou $\alpha \in (0,1)$ est ***a completer apres execution*** selon le compromis reactivite/stabilite.

12.2.4 Distance heuristique : relation inverse a la taille apparente

Sans calibration, la distance est estimee de maniere relative. Une heuristique courante relie la distance d a une taille en pixels (largeur ou hauteur de bounding box) p :

$$d \propto \frac{1}{p}$$

Une forme parametrique possible :

$$d_{\text{heur}} = \frac{k}{p + \varepsilon}$$

avec k et ε ***a completer apres execution***. La quantification en zones (safe/close/-very_close) repose sur des seuils d_1, d_2 ***a completer apres execution***.

12.2.5 Score de risque : combinaison ponderee et saturation

Le score (0–100) sert d'indicateur graduel, puis est transforme en classes SAFE/WARNING/DANGER. Une forme generale :

$$score = \text{clip}_{[0,100]} \left(w_{\text{voie}} S_{\text{voie}} + w_{\text{dist}} S_{\text{dist}} + w_{\text{ctx}} S_{\text{ctx}} \right)$$

Les sous-scores $S_{\cdot}$ representent des signaux interpretable (ex. deviation de voie, zone de distance, contexte), et les poids $w_{\cdot}$ sont ***a completer apres execution*** (differents selon MODE city/highway).

12.2.6 Logique evenementielle : anti-spam audio (cooldown)

Le module audio est un exemple de modelisation par etat. Soit t le temps courant et t_{last} le temps de la derniere alerte. Une alerte est autorisee si :

$$t - t_{\text{last}} \geq T_{\text{cooldown}}$$

avec T_{cooldown} ***a completer apres execution***. La priorite danger impose une regle supplementaire : un danger peut preempter un warning selon une politique definie (etat courant, temps ecoule, priorite).

12.3 Ouverture (V2) : calibration pinhole et metriques ADAS

Cette partie est une perspective (pas un resultat V1). Avec une calibration camera (modele pinhole), une distance plus credible devient possible. Le principe :

$$Z \approx \frac{f \cdot H}{h}$$

ou Z est la distance, f la focale (en pixels), H une dimension reelle de reference, h sa taille apparente en pixels. Tous les parametres sont ***a completer apres execution*** et demandent un protocole de mesure.

Une fois une distance estimee, deux metriques ADAS courantes deviennent envisageables :

$$\text{Time headway} = \frac{Z}{v}, \qquad \text{TTC} = \frac{Z}{\Delta v}$$

avec v (vitesse) et Δv (vitesse relative) ***a completer apres execution*** selon les capteurs disponibles (ou estimation video en V2).

Pourquoi ces formules renforcent la credibilite du projet

Elles montrent une trajectoire d'evolution : de l'heuristique interpretable (V1) vers une estimation plus physique et des metriques ADAS (V2), avec une methode de calibration et de validation.

13 Progression personnelle

13.1 Point de depart : un projet de reconnaissance faciale (image -> decision)

Avant DriveGuardian IA, j'ai travaille sur un projet de reconnaissance faciale. Ce type de projet m'a permis de consolider les bases de la vision par ordinateur (pretraitement, detection, descripteurs/embeddings selon approche, evaluation), mais il reste souvent centre sur une tache relativement **ponctuelle** : a partir d'une image (ou d'une sequence courte), produire une decision (identite / verification).

Cette experience m'a donne une base solide, mais j'ai voulu franchir une etape : passer d'un modele « local » a un **systeme complet** traite comme un mini-produit, avec un pipeline, une interface, des etats, et des livrables.

13.2 Le saut realise avec DriveGuardian IA : video continue + logique systeme

DriveGuardian IAm'a fait changer d'echelle :

— **Temporalite** : une frame n'a pas de sens seule ; c'est la coherence sur la duree qui compte.

— **Multi-modules** : voie, vehicules, distance heuristique, risque, audio, dashboard, exports.

— **Robustesse** : un prototype doit survivre aux erreurs (fichiers, encodage, formats), pas seulement fonctionner une fois.

— **UX** : une information juste mais illisible est inutile ; j'ai appris a hierarchiser et simplifier.

Ce que j'ai appris

Le coeur du projet n'est pas uniquement l'algorithme : c'est la capacite a transformer des signaux imparfaits en une assistance **stable, interpretable et utile**.

13.3 Ameliorations concretes realisees pendant le developpement

13.3.1 1) Robustesse d'execution (fichiers, formats, erreurs)

J'ai rencontre et resolu des problemes typiques d'un projet appliqué :

— ressources manquantes (ex. detecteur `cars.xml`, fichiers audio) : verification des chemins, messages d'erreur clairs ;

— differences de formats (mp3 vs wav) : standardisation et controle de compatibilite ;

— stabilite globale : eviter un comportement « fragile » qui casse selon la machine ou l'arborescence.

Ce travail m'a appris a raisonner en **conditions reelles d'utilisation** et pas seulement en contexte de developpement.

13.3.2 2) Stabilite temporelle (lissage, persistance, anti-flicker)

Le passage a la video m'a oblige a traiter un point essentiel : **la stabilite percue**. Une sortie qui change brutalement a chaque frame degrade la confiance et l'utilite. J'ai donc integre des mecanismes de stabilisation (moyenne glissante, persistance d'etat, hysteresis logique si necessaire), afin de produire des indicateurs plus coherents.

13.3.3 3) Logique evenementielle (audio anti-spam + priorite danger)

Un point marquant du projet est la gestion de l'audio. Jouer un son n'est pas difficile ; ce qui est difficile est d'alerter correctement :

— eviter la repetition excessive (fatigue, spam) ;

— garantir qu'un danger n'est pas masque par des warnings ;

— gerer l'etat : derniere alerte, temps associe, priorites, options ON/OFF.

Cette partie m'a fait progresser sur les concepts de **machine a etats**, de **priorites** et de **systeme temps reel**.

13.3.4 4) Presentation professionnelle (dashboard + bilan fin de trajet)

J'ai volontairement travaille l'aspect « livrable » :

— dashboard plus propre : hierarchie claire (risque d'abord, puis voie/vehicules, puis details) ;

— texte explicatif « smart » : expliquer la cause dominante du risque ;

— fin de trajet : export CSV + graphes PNG + bilan et recommandations.

Cette evolution transforme une demonstration technique en un outil d'analyse, coherent avec les attentes d'un jury.

13.3.5 5) Qualite d'affichage : problemes Unicode (accents) et correction

Le probleme des accents affiches en « ?? » dans OpenCV m'a pousse a chercher une solution robuste (normalisation Unicode), preuve d'une attention au detail et a la qualite de restitution.

13.4 Organisation de travail : iteration et documentation

Sur DriveGuardian IA, j'ai adopte une methode d'iteration courte :

1. integrer une fonctionnalite minimale ;
2. observer sur une sequence representative ;
3. corriger les erreurs et stabiliser ;
4. ameliorer la lisibilite et la coherence ;
5. produire une sortie exploitable (logs, exports, figures).

Cette methode m'a aide a progresser sans bloquer le projet sur une seule partie trop longtemps, tout en gardant une trajectoire vers un resultat presentable.

Progression d'ingenieur

Je suis passe d'un objectif « faire fonctionner » a un objectif « faire fonctionner de maniere fiable, lisible, et defendable ».

13.5 Ce que cette progression dit de mon profil

Ce projet met en evidence une evolution vers des competences attendues en ecole d'ingenieurs :

— raisonner en compromis (precision vs robustesse vs lisibilite) ;
— concevoir une architecture modulaire et evolutive ;
— gerer des contraintes temps reel et de restitution ;
— documenter et produire des livrables (bilan, exports, figures).

13.6 Perspectives : comment je veux aller plus loin (V2)

La V1 valide une base interpretable. La V2 vise a renforcer la credibilite technique :

— calibration physique (modele pinhole) pour une distance plus fiable ;

— metriques ADAS (time headway, TTC) lorsque les elements necessaires seront disponibles ;

— remplacement progressif de modules par du deep learning (YOLO / segmentation voie) ;

— campagne de tests sur captations personnelles (conditions variees).

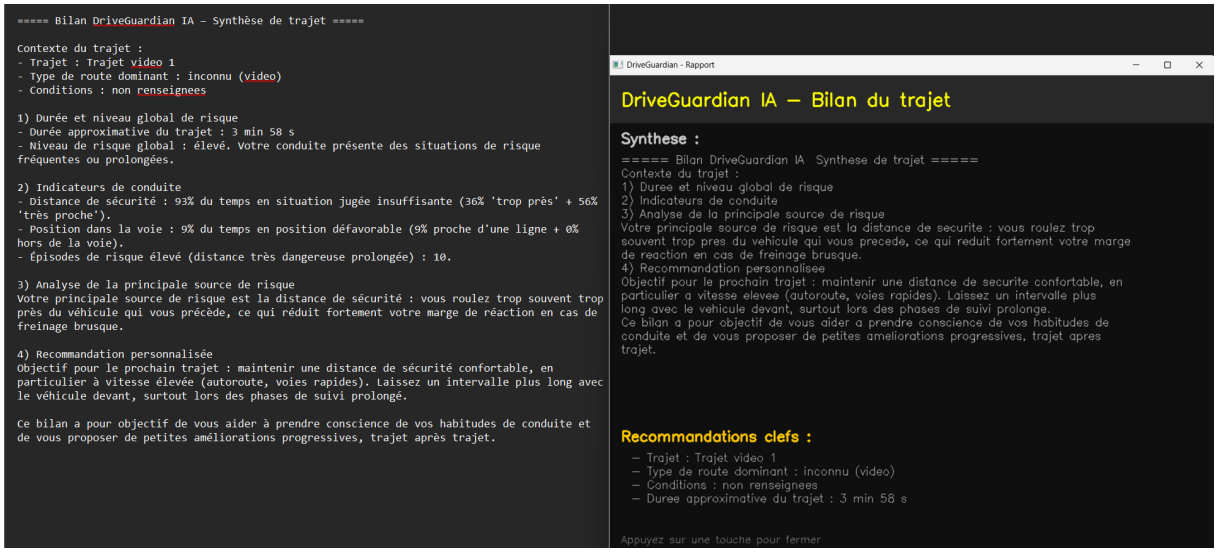


FIGURE 13.1 – Evolution du score de risque .

14 Calibration physique de distance (V2)

14.1 Pourquoi calibrer ?

En V1, la distance est heuristique. Une calibration permettrait de produire une estimation plus credible, et surtout de calculer des metriques avancees (time headway, TTC) utiles en ADAS.

14.2 Modele pinhole (concept)

Le modele pinhole relie taille apparente en pixels, focale et distance. Toute application numerique est *a completer apres execution*(depend de la camera et de mesures).

14.3 Protocole simple

Fixer la camera, mesurer des distances connues, mesurer les tailles en pixels, ajuster un modele, puis valider sur scenes differentes.

Precautions

Un changement d'inclinaison ou de zoom invalide une calibration : documentation et reproductibilite sont essentielles.

15 Limites et perspectives

15.1 Limites V1 (assumees et documentees)

— methodes classiques sensibles (voie) ;
— Haar moins robuste que des detecteurs modernes ;
— distance non metrique ;
— clignotants uniquement indicatifs.

15.2 Perspectives V2

Evolutions

— deep learning (YOLO, segmentation) ;
— calibration : distance plus fiable, time headway, TTC ;
— tests reels : nuit/meteo, vibrations, variations camera ;
— modularisation (config, tests, logs) et evaluation multi-videos.

16 Conclusion

DriveGuardian IApropose une chaine complete d'assistance dashcam : perception, risque, alertes, dashboard et post-analyse. La valeur du projet repose autant sur les algorithmes que sur la demarche d'ingenierie : robustesse, stabilite, interpretabilite, et production de livrables exploitables. Une V2 (calibration + deep learning + tests reels) constitue une evolution naturelle.

A Tableaux a completer apres execution

TABLE A.1 – Seuils et ponderations (*a completer apres execution*).

Element	**Valeur**
Seuils offset voie	`centre si abs(offset)<0.15 ; proche ligne si 0.15<=abs(offset)<0.30 ; hors voie si abs(offset)>=0.30.`
Seuils zones distance	`city: safe si d>25 ; close si 12<d<=25 ; very_close si d<=12 ; highway: safe si d>40 ; close si 20<d<=40 ; very_close si d<=20.` ***(estimation V1, a affiner par tests)***
Cooldown warning	`WARNING_MIN_GAP=3.5 s`
Cooldown danger	`DANGER_MIN_GAP=2.5 s ; DANGER_OVERRIDE_GAP=0.3 s`
Ponderations MODE city	`w_dist=0.45, w_lane=0.30, w_stab=0.15, w_trafic=0.10`
Ponderations MODE highway	`w_dist=0.55, w_lane=0.20, w_stab=0.15, w_trafic=0.10`

B Pseudo-code global

Algorithm 1: Pseudo-code global de DriveGuardian IA.

Input: Flux video dashcam

Output: Dashboard temps reel, exports fin de trajet

Initialiser parametres (MODE, DEMO_MODE, options audio)

Charger ressources (cars.xml, warning.wav, danger.wav)

Initialiser etats (historiques, logs)

while *frame disponible* **do**

- Lire frame
- Pretraiter (ROI, filtres)
- Detecter voie (Canny + Hough + offset + lissage)
- Detecter vehicules (cascade + top-3 + pseudo-radar)
- Estimer distance (heuristique + zones)
- Detecter clignotants (approx, optionnel)
- Evaluer risque (SAFE/WARNING/DANGER + score + texte)
- Jouer audio (priorite danger + cooldown anti-spam)
- Afficher dashboard
- Enregistrer metriques (buffer CSV)

Exporter CSV

Generer graphes PNG

Afficher bilan + recommandations

C Reference

— Video de test V1 : https://www.youtube.com/watch?v=XEzXUjzXF3s&t=767s