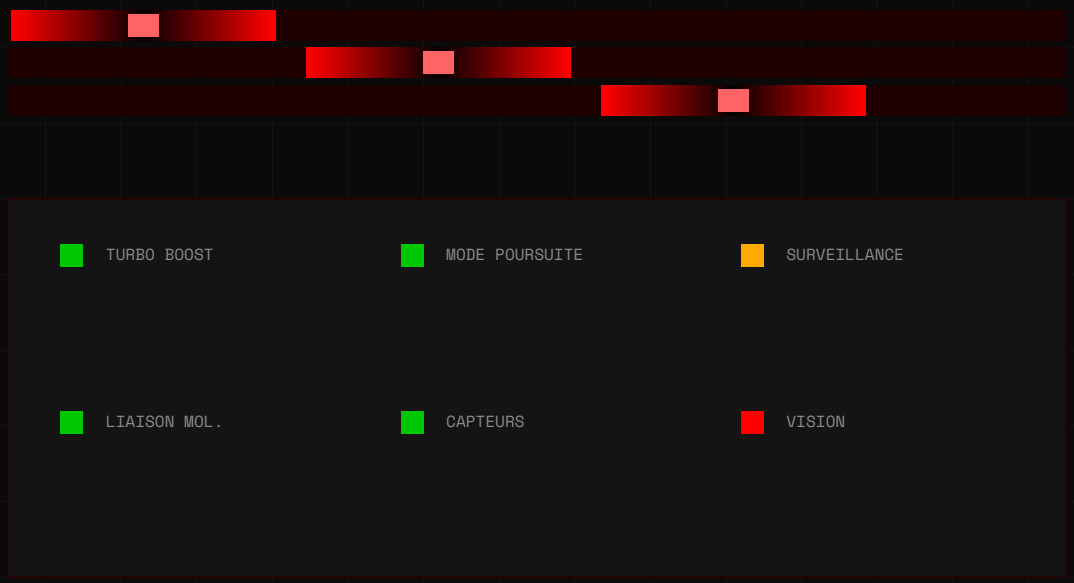
KITT

KNIGHT INDUSTRIES TWO THOUSAND

MODE D'EMPLOI

Intelligence Artificielle Vocale Multilingue

NVIDIA Jetson Orin Nano Super



Version 3.0

18 février 2026

CRÉÉ PAR MANIX — EMMANUEL GELINNE

on3egs@icloud.com | KITT Franco-Belge

TABLE DES MATIÈRES

1. PRÉSENTATION

KITT (Knight Industries Two Thousand) est une intelligence artificielle vocale embarquée au style rétro-futuriste, inspirée de la série Knight Rider, conçue en Belgique par Manix (Emmanuel Gelinne), fondateur du groupe KITT Franco-Belge.

Ce projet transforme un NVIDIA Jetson Orin Nano Super en un assistant vocal interactif : sophistiqué, articulé, avec un humour sec et un esprit vif. Il répond en plusieurs langues (français, anglais, allemand, italien, portugais) avec une voix synthétique robotique adaptée à chaque langue.

KITT s'intègre dans un tableau de bord automobile IoT avancé avec composants ZA Elettronica (switchpods, scanner, voicebox, écrans) — dans l'esprit du Knight 2000 IoT de Mario Ravasi.

> Capacités

- Conversation vocale multilingue (fr/en/de/it/pt) avec personnalité KITT unique
- STT GPU : Whisper base CUDA float16 (~300ms) avec détection automatique de la langue
- TTS GPU : Piper multilingue CUDA ~490ms avec effets robotiques SoX (pitch, overdrive)
- LLM local Qwen 2.5 3B sur GPU (~1400ms) — aucune connexion internet requise
- Vision par caméra : détection d'objets YOLOX-S, couleurs, vêtements (~580ms)
- 4 modes d'écoute : Push-to-Talk, Auto-VAD, Wake Word, Caméra
- Wake word « KITT » : activation vocale sans contact
- Monitor WebSocket temps réel (conversations en direct)
- Client terminal vocal (terminal_chat.py)
- Accès distant sécurisé via tunnel Cloudflare
- Mémoire persistante entre sessions (faits utilisateur)
- Reconnaissance faciale du conducteur (YuNet + SFace)
- Contexte IoT embarqué : capteurs temps réel, modules distribués
- Latence totale ~2,3 secondes — réponse vocale complète

NOTE IMPORTANTE

KITT fonctionne entièrement hors ligne.
Aucune donnée n'est envoyée vers le cloud.
Votre vie privée est protégée par le réseau neural adaptatif... numériquement parlant.

2. PRÉREQUIS

> Matériel requis

- NVIDIA Jetson Orin Nano Super (8 Go de RAM)
- Carte microSD 64 Go minimum (128 Go recommandé)
- Webcam USB compatible V4L2 (pour la vision)
- Microphone USB ou jack (pour les commandes vocales)
- Haut-parleur ou sortie audio (pour la voix de l'IA)
- Connexion réseau local (Wi-Fi ou Ethernet)
- Alimentation 5V/4A (fournie avec le Jetson)

> Logiciel de base

Composant	Version
JetPack	6.2.2
Ubuntu	22.04.5 LTS
CUDA	12.6
Python	3.10
OpenCV	4.8.0 (système)
Noyau Linux	5.15.185-tegra

> Paquets système nécessaires

```
sudo apt update
sudo apt install -y ffmpeg sox libportaudio2 \
    curl build-essential python3-pip
```

3. INSTALLATION

> Étape 1 : Cloner le projet

```
cd /home/kitt
git clone <dépôt> kitt-ai
cd kitt-ai
```

> Étape 2 : Environnement Python

```
python3 -m venv venv
source venv/bin/activate
pip install aiohttp faster-whisper langdetect
```

> Étape 3 : Compiler llama.cpp

```
cd /home/kitt/llama.cpp
cmake -B build \
    -DGGML_CUDA=ON \
    -DCMAKE_CUDA_ARCHITECTURES=87 \
    -DCMAKE_BUILD_TYPE=Release
cmake --build build -j$(nproc)
```

> Étape 4 : Installer la vision

```
cd /home/kitt/kitt-ai
bash install_vision.sh
```

Ce script télécharge le modèle YOLOv8n au format ONNX (~13 Mo) dans le dossier models/.

> Étape 5 : Modèles TTS multilingues

Télécharger les modèles Piper supplémentaires depuis HuggingFace dans le dossier models/ :

- fr_FR-tom-medium.onnx (inclus, CUDA)
- en_US-lessac-medium.onnx
- de_DE-thorsten-medium.onnx
- it_IT-paola-medium.onnx
- pt_BR-faber-medium.onnx

> Étape 6 : Service de reconnaissance faciale

```
sudo cp driver/kitt-recognition.service \
    /etc/systemd/system/
sudo systemctl daemon-reload
sudo systemctl enable kitt-recognition.service
```

4. DÉMARRAGE

Le démarrage de KITT s'effectue en une seule commande qui lance à la fois le serveur LLM et le serveur web.

> Lancement normal

```
cd /home/kitt/kitt-ai
bash start_kyronex.sh
```

Le script démarre automatiquement :

1. llama-server sur le port 8080 (modèle LLM sur GPU)
2. Le serveur KITT sur le port 3000 (interface web)

> Mode tunnel (accès distant Cloudflare)

```
TUNNEL=1 bash start_kyronex.sh
```

Active un tunnel Cloudflare sécurisé. L'URL publique est affichée dans /tmp/cloudflared.log. Un mot de passe est requis (défaut : 1982, modifiable via KYRONEX_PASSWORD).

> Client terminal (optionnel)

```
# Dans un autre terminal :
venv/bin/python3 terminal_chat.py
```

Client vocal en ligne de commande. Entrée seule = micro push-to-talk. Tapez 'auto' pour le mode écoute continue VAD.

> Monitor temps réel (optionnel)

```
# Dans un autre terminal :
venv/bin/python3 monitor.py
```

Fenêtre tkinter affichant les conversations en direct via WebSocket. Reconnexion automatique toutes les 5 secondes.

> Accès à l'interface

ADRESSES D'ACCÈS

Depuis le Jetson : https://localhost:3000
Depuis le réseau : https://192.168.1.4:3000

Remplacez l'IP si nécessaire (vérifier avec :
ip addr show | grep '192.168')

> Vérification

Lorsque KITT est opérationnel, vous verrez dans le terminal :

```
==========================================
  KITT - Knight Industries Two Thousand
  Serveur IA Vocal - Jetson Orin Nano Super
==========================================
[OK] Whisper prêt (CUDA float16)
[OK] TTS multilingue prêt (fr/en/de/it/pt)
  HTTPS actif - https://localhost:3000
```

L'indicateur en bas de l'interface web affichera « KITT EN LIGNE » en vert.

5. INTERFACE WEB

L'interface web adopte un design rétro-futuriste noir et rouge, évoquant les tableaux de bord high-tech des années 80.

> Éléments de l'interface

INDICATEUR LUMINEUX

La barre rouge oscillante en haut de l'écran
est l'indicateur d'activité de KITT.
Elle change de comportement selon l'état :

- Oscillation lente : en veille
- Oscillation rapide : KITT parle
- Orange : écoute en cours
- Vert : auto-écoute active

> 4 modes d'écoute

Bouton	Fonction
MIC (rouge)	Push-to-talk : cliquer pour parler, cliquer pour arrêter
AUTO (vert)	Écoute continue VAD — détecte automatiquement la parole
WAKE (violet)	Wake word : écoute passive, réagit uniquement sur 'KITT'
CAM (bleu)	Capture caméra + analyse visuelle YOLOX-S
ENVOYER	Envoie le message texte saisi dans le champ

> Saisie de texte

Tapez votre message dans le champ de saisie et appuyez sur Entrée ou cliquez sur ENVOYER. KITT répondra en texte (streaming en temps réel) puis en audio dans la langue détectée automatiquement.

> Indicateurs de performance

Sous chaque réponse de KITT, vous verrez les temps de traitement : LLM (réflexion), TTS (synthèse vocale), et Vision (si la caméra a été utilisée).

6. COMMANDES VOCALES

> Mode Push-to-Talk

Cliquez sur le bouton microphone (rond rouge). Il devient rouge vif et pulse pendant l'enregistrement. Parlez, puis cliquez à nouveau pour arrêter. La transcription est automatique (Whisper) avec détection automatique de la langue.

> Mode Auto-écoute (VAD)

Cliquez sur le bouton AUTO (vert). KITT écoute en permanence et détecte automatiquement quand vous parlez. Seuil de détection : 0.008. Silence : 800ms. Parole minimum : 500ms.

ANTI-ÉCHO

En mode auto-écoute, KITT se met en sourdine pendant qu'il parle et 1,5 seconde après, pour éviter de s'écouter lui-même.

> Mode Wake Word

Cliquez sur le bouton WAKE (violet). KITT écoute en continu de façon passive et ne réagit que si vous prononcez le mot d'activation « KITT ». Une fois détecté, il passe en mode écoute active pendant 6 secondes pour capter votre question. Utile pour une utilisation mains-libres en conduite.

> TTS Multilingue automatique

KITT détecte automatiquement la langue de chaque message (via langdetect) et répond avec le modèle vocal correspondant. Vous pouvez aussi forcer la langue via le paramètre 'lang' dans l'API. Parlez en anglais, KITT vous répond en anglais.

> Exemples de phrases

- « Bonjour KITT, comment vas-tu ? » (fr)
- « Hello KITT, what do you see? » (en)
- « KITT, was siehst du? » (de)
- « Active le Turbo Boost ! »
- « Qu'est-ce que tu vois ? » (déclenche la vision)
- « Qu'est-ce que je porte ? » (déclenche la vision)
- « Regarde devant toi. » (déclenche la vision)

7. VISION PAR CAMÉRA

KITT peut « voir » grâce à la webcam connectée au Jetson. Il utilise le modèle YOLOX-S (Apache 2.0, Megvii) via cv2.dnn pour détecter les objets et analyser les couleurs dans la scène. Inference : ~580ms, 80 classes COCO.

> Utilisation manuelle

Cliquez sur le bouton caméra (icône appareil photo). Le bouton devient bleu pendant la capture. KITT capture une image, l'analyse, et répond en décrivant ce qu'il voit.

Vous pouvez taper une question avant de cliquer :
« Qu'est-ce que je tiens ? » → clic caméra
Si le champ est vide, KITT utilisera « Que vois-tu ? »

> Détection automatique

Lorsque vous posez une question contenant certains mots-clés, KITT active automatiquement la caméra. Cela fonctionne aussi en mode vocal.

MOTS-CLÉS DE VISION

vois, regarde, tiens, porte, portes, couleur,
devant toi, caméra, montre, habillé, vêtement

> Capacités de détection

- 80 catégories d'objets (personnes, animaux, objets courants)
- Analyse des couleurs dominantes de chaque objet
- Pour les personnes : couleur du haut et du bas du corps
- Résultat transmis au LLM pour une réponse naturelle

> Exemple de dialogue

```
Vous : "Qu'est-ce que tu vois ?"

KITT : "Mes capteurs visuels détectent
  une personne portant un haut bleu, ainsi
  qu'une tasse à proximité."
```

8. RECONNAISSANCE FACIALE

KITT peut identifier son conducteur grâce à la reconnaissance faciale. Cette fonctionnalité utilise les modèles YuNet (détection) et SFace (reconnaissance) d'OpenCV.

> Enregistrer un visage

```
cd /home/kitt/kitt-ai/driver
python3 recognition.py --enroll MonNom
```

Placez votre visage devant la caméra. Le système capture 5 images sous différents angles pour créer votre profil. Le fichier est sauvegardé dans driver/faces/MonNom.npy

> Lancer la reconnaissance

```
# Manuellement :
python3 recognition.py

# Via le service systemd :
sudo systemctl start kitt-recognition
```

Lorsqu'un conducteur est reconnu (3 correspondances consécutives), KITT le salue vocalement et démarre le système.

> Service automatique

Le service kitt-recognition.service peut être activé au démarrage du Jetson pour une reconnaissance automatique :

```
sudo systemctl enable kitt-recognition
sudo systemctl start kitt-recognition

# Consulter les logs :
journalctl -u kitt-recognition -f
```

9. ARCHITECTURE TECHNIQUE

> Vue d'ensemble

```
Navigateur / Terminal (port 3000 HTTPS)
  |
  +-- Texte  --> /api/chat/stream --> LLM (~1400ms)
  +-- Voix   --> /api/stt --> Whisper CUDA (~300ms)
  +-- Vision --> /api/vision --> YOLOX-S (~580ms)
  |
  +-- Audio  <-- MultilingualTTS CUDA + SoX (~490ms)
  |              (fr CUDA + en/de/it/pt CPU LRU)
  |
  +-- Monitor WebSocket /api/monitor/ws
  +-- llama-server (port 8080, GPU CUDA)
  +-- Tunnel Cloudflare (mode TUNNEL=1)
```

> Composants

Composant	Détail
Serveur web	Python aiohttp, port 3000 HTTPS
LLM	Qwen 2.5 3B (Q5_K_M), llama.cpp
STT	faster-whisper base, CUDA float16, lang=auto
TTS fr	Piper fr_FR-tom-medium, CUDA (permanent)
TTS autres	Piper en/de/it/pt, CPU, LRU cache x1
Détection langue	langdetect (~3ms, seed=0)
Vision	YOLOX-S ONNX, cv2.dnn (Apache 2.0, ~580ms)
Reconnaissance	YuNet + SFace (OpenCV)
GPU	CUDA 12.6, architecture SM 87

> Paramètres llama-server (optimisés anti-OOM)

Paramètre	Valeur
--n-gpu-layers	99 (tout sur GPU)
--ctx-size	1024 (réduit vs 2048)
--batch-size	512
--threads	4
--flash-attn on	Flash Attention activé

> Ports réseau

Port	Service

3000	Serveur KITT (HTTPS)
8080	llama-server (API LLM locale)

> Arborescence des fichiers

```
kitt-ai/
  kyronex_server.py      # Serveur principal KITT
  piper_gpu.py           # TTS GPU + MultilingualTTS
  terminal_chat.py       # Client terminal vocal
  monitor.py             # Monitor WebSocket tkinter
  vision.py              # Vision YOLOX-S
  start_kyronex.sh       # Script de démarrage
  static/index.html      # Interface web PWA
  models/                # Modèles IA (LLM + TTS + ONNX)
  audio_cache/           # Cache audio temp.
  driver/                # Reconnaissance faciale
  certs/                 # Certificats HTTPS
  logs/                  # Journaux JSONL conversations
  site/                  # Site web GitHub Pages
```

10. DÉPANNAGE

> KITT ne démarre pas

- Vérifiez que les paquets sont installés : ffmpeg, sox, libportaudio2
- Vérifiez le venv : source venv/bin/activate && pip list
- Consultez les logs : bash start_kyronex.sh (dans le terminal)
- Vérifiez LD_LIBRARY_PATH si erreur libctranslate2.so.4

> LLM HORS LIGNE dans l'interface

- llama-server n'a pas fini de charger le modèle (patience ~30s)
- Vérifiez : curl http://localhost:8080/health
- Le modèle GGUF est-il présent dans models/ ?

> Le microphone ne fonctionne pas

- Autorisez l'accès au micro dans votre navigateur
- HTTPS est requis pour le micro (sauf localhost)
- Vérifiez : arecord -l (liste les micros détectés)

> La caméra ne fonctionne pas

- Vérifiez : ls /dev/video* (doit afficher video0)
- Testez : python3 vision.py --test
- Assurez-vous qu'un seul programme utilise la caméra

> La voix est absente ou déformée

- Vérifiez la sortie audio : aplay /usr/share/sounds/alsa/Front_Center.wav
- Vérifiez que le modèle TTS existe dans models/
- Vérifiez SoX : sox --version

> Surveiller la mémoire (anti-OOM)

```
# VRAM en temps réel
tegrastats | grep -o 'RAM [0-9]*/[0-9]*MB'

# Log VRAM KITT
tail -f /tmp/kitt_vram.log

# Test LLM
curl http://localhost:8080/health

# État services
sudo systemctl status kitt-recognition
```

ALERTE OOM

Si LFB (Largest Free Block) < 4MB dans tegrastats :
fragmentation critique. Redémarrer llama-server.
ctx-size 1024 + flash-attn évitent ce problème.

11. CRÉDITS ET LICENCE

CRÉATEUR

Manix

(Emmanuel Gelinne)

> Contact

E-mail	on3egs@icloud.com
Groupe	KITT Franco-Belge (Facebook)
Site web	https://on3egs.github.io/manix-kitt/
Rôle	Fondateur & Administrateur

> Licence

ELASTIC LICENSE 2.0

Copyright (c) 2026 Manix (Emmanuel Gelinne)

Utilisation libre pour usage personnel, éducatif
et non-commercial autorisée.

INTERDIT sans accord écrit de l'auteur :
- Offrir le logiciel comme service commercial
- Retirer ou modifier les notices de licence
- Créer des produits dérivés commerciaux

L'auteur conserve tous les droits commerciaux.
Licence complète : fichier LICENSE du projet.

> Technologies utilisées

- NVIDIA Jetson Orin Nano Super — plateforme matérielle
- llama.cpp — inférence LLM optimisée GPU
- Qwen 2.5 3B Instruct — modèle de langage
- Piper TTS — synthèse vocale neuronale multilingue
- faster-whisper — reconnaissance vocale CUDA
- langdetect — détection automatique de la langue
- YOLOX-S (Megvii, Apache 2.0) — détection d'objets
- OpenCV — vision par ordinateur
- SoX — effets audio robotiques

- Python aiohttp — serveur web asynchrone
- ZA Elettronica — composants IoT embarqués

> Remerciements

Merci à la communauté du groupe KITT Franco-Belge et à Mario Ravasi (Knight 2000 IoT) pour leur inspiration et leur soutien.

« La technologie au service de l'humain. »
— Manix