Robotique, Computer Vision

Thomas Aurélien Enzo

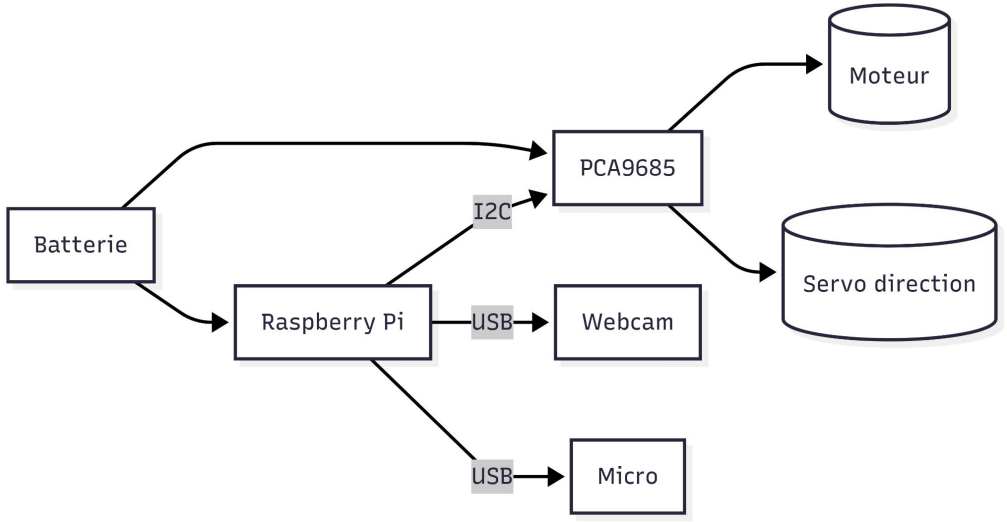
Présentation du système

- Système embarqué pour transformer une voiture RC en véhicule autonome

- Principales fonctionnalités :
  - Reconnaissance vocale (“avance”)
  - Vision par ordinateur (triangle rouge)
  - Contrôle moteur/servo via PWM
  - Accès à distance via SSH

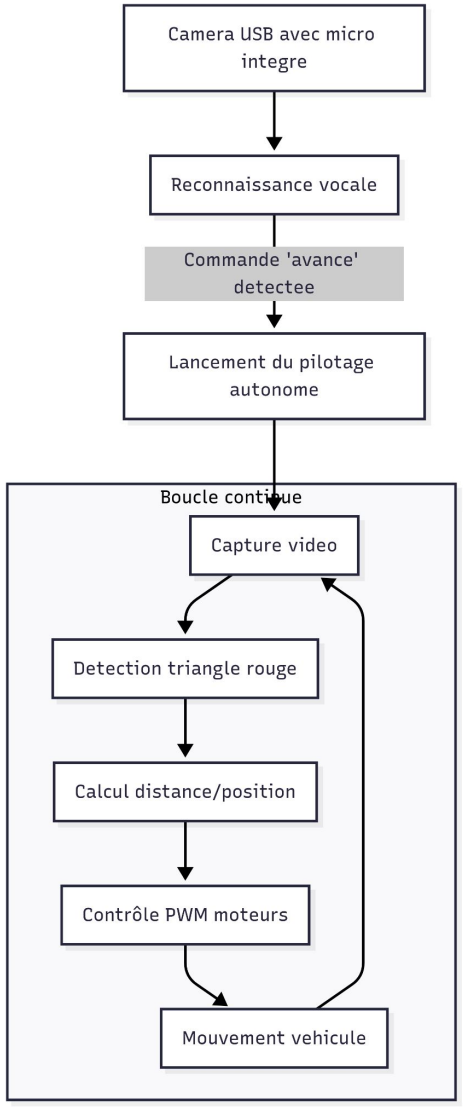
- Flux Opérationnel :
  - Micro → “avance” → Caméra → Triangle détecté → Calcul distance → PWM → Mouvement

Architecture matérielle



Composant	Rôle
Raspberry Pi 4	Calcul, héberge les scripts
PCA9685	Contrôle PWM des moteurs
Caméra USB	Vision pour la détection
Micro intégré	Déclenchement vocal
Batterie 5V/3A	Alimentation Raspberry
Batterie LiPo	Alimentation voiture

Architecture Logicielle



- Pipeline global
  - Acquisition vidéo (OpenCV, 736×414 @~50 FPS)
  - Détection de triangle rouge (HSV + contours)
  - Estimation distance via perspective simple
  - Commande PWM (throttle & steering)

Architecture Logicielle (Zoom sur le code)

Extraits clés de `estimate_distance()` et Lissage Proportionnel pour éviter les oscillations

```python
def estimate_distance(triangle_width_px, known_width_cm=5.0, focal_length_px=240):
    try:
        return round((known_width_cm * focal_length_px) / triangle_width_px, 2)
    except ZeroDivisionError:
        return -1
```

```python
# --- Moyennes glissantes ---
throttle_history = deque([NEUTRAL] * 3, maxlen=3)
steering_history = deque([NEUTRAL] * 1, maxlen=1)

def smooth(val, history):
    history.append(val)
    return int(np.mean(history))
```

Configuration audio

- Liste et sélection du micro via sounddevice*
- Test via : test_micro(device_index)
- Utilisation de PocketSphinx :
  - Mot-clé : "avance"
  - Seuil : `1e-20`

```
# 3) Configuration pour le keyword spotting français
MODEL_PATH = '/usr/share/pocketsphinx/model/fr-fr-ptm-5.2'
DICT_PATH  = '/usr/share/pocketsphinx/model/fr-fr-ptm-5.2/fr.dict'
speech = LiveSpeech(
    hmm=MODEL_PATH,
    dict=DICT_PATH,
    keyphrase='avance',
    kws_threshold=1e-20,
    audio_device=MIC_DEVICE_INDEX,
    buffer_size=2048,
    full_utt=False
)

# 4) Boucle d’écoute
print("\nEn attente de « avance »… (Ctrl+C pour quitter)")
for phrase in speech:
    print("✅ Phrase détectée : « {} »".format(phrase))
    subprocess.run(["./1.py"])
    break
```

Contrôle des moteurs

- Contrôle PWM :
  - Valeurs définies (NEUTRAL, FORWARD, LEFT, RIGHT)

- Algorithme :
  - Throttle proportionnel à la distance
  - Steering ajusté selon la vitesse

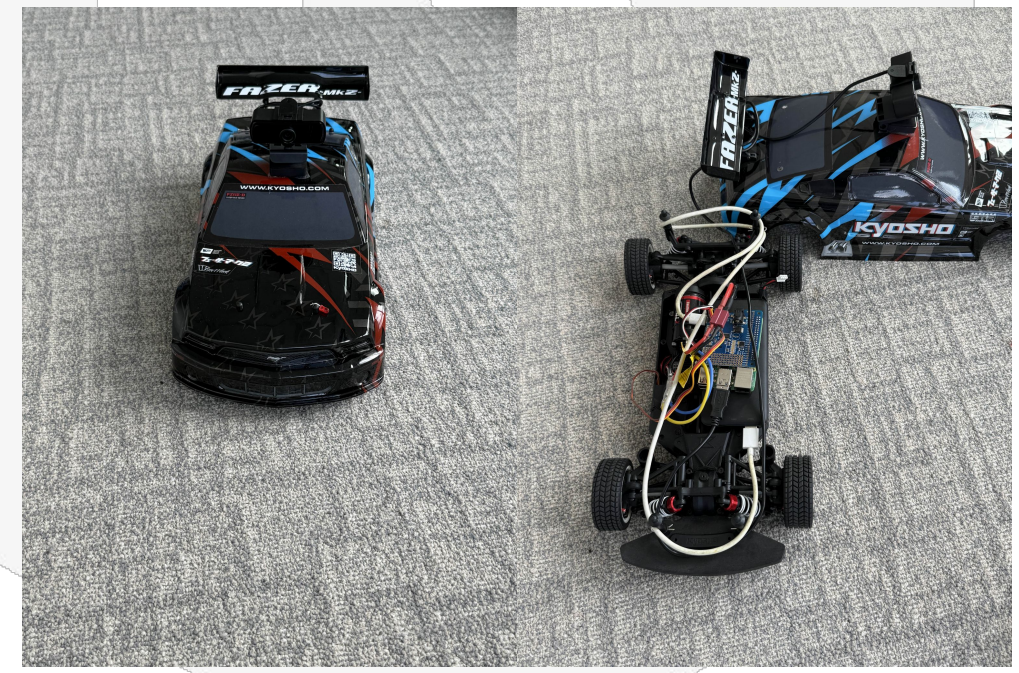
Journalisation & Débogage

- Surveillance via SSH :
  - `htop`, `i2cdetect`, `libcamera-hello`
- Lancement : 1.py
- Affichage en direct : throttle + direction

Améliorations futures

- Eviter des obstacles
- Intégrer un télémètre laser
- Interface Web pour contrôler chaque mode et pour contrôler la voiture a distance

Phase de Test



hexagone

École Supérieure d'Informatique

Merci pour votre attention

La Team ToCar