

# FetchBot: Build your intelligent Mars Rover

---

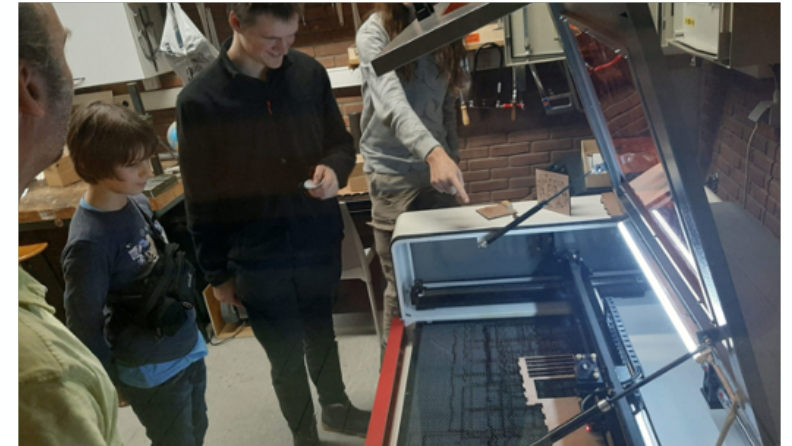
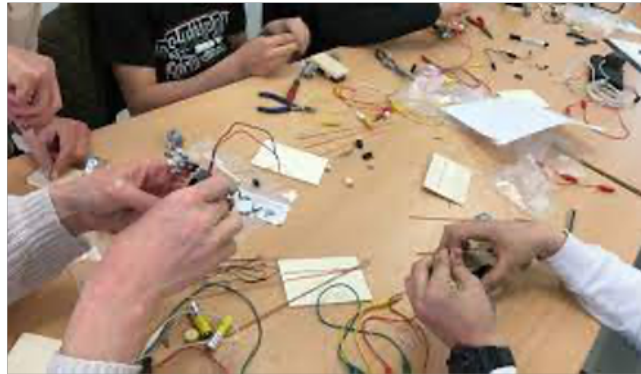
Yann-Aël Le Borgne, 26/08/2022

La Scientothèque and ESERO Belgium

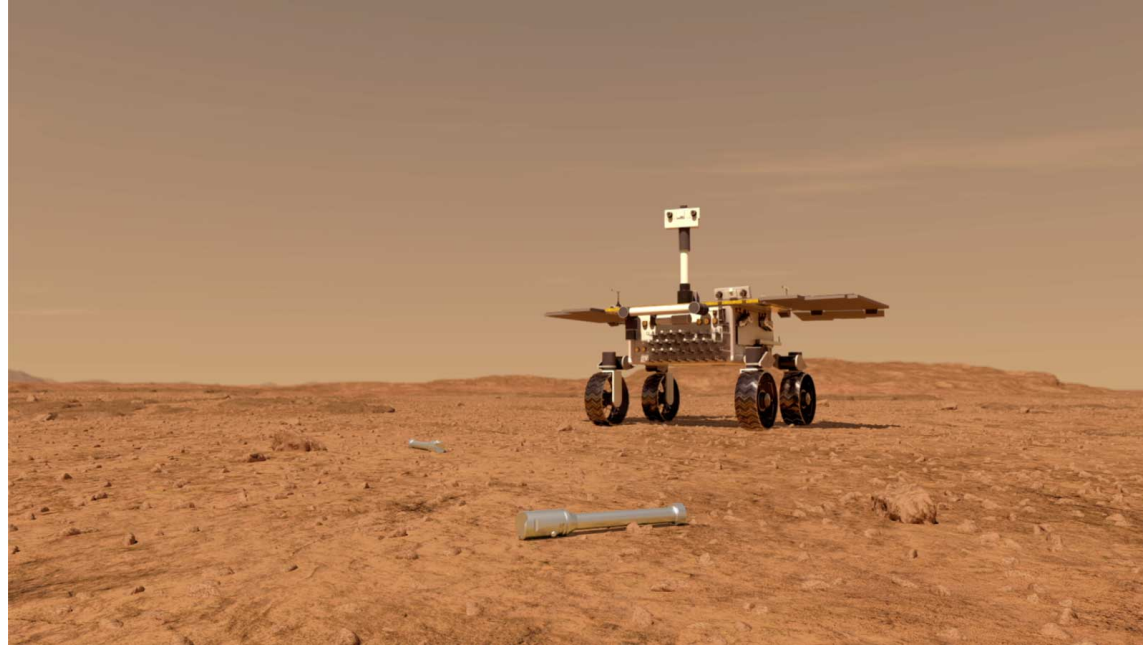
# La Scientothèque

L'ÉGALITÉ DES CHANCES PAR LES SCIENCES

*Equal opportunities through sciences*

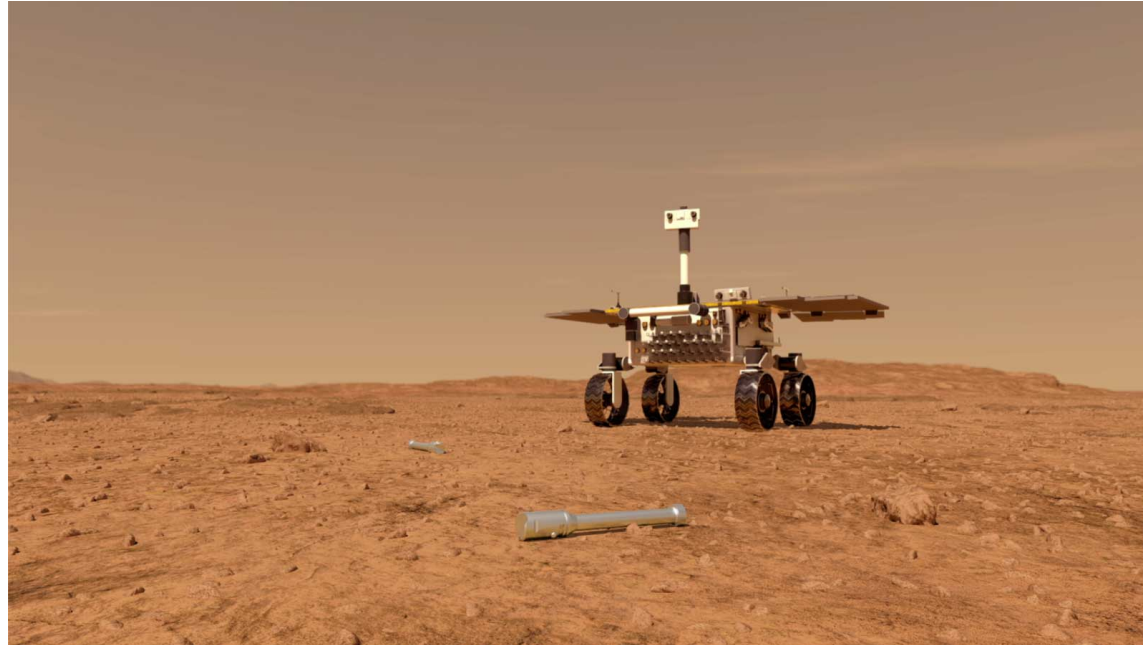


# FetchBot inspiration: Mars exploration - Fetch mission



1. Create an AI for image recognition and use it in a Scratch ou Python program
2. Build a Mars rover FetchBot
3. Program the FetchBot to find tubes on a Mars terrain

# FetchBot inspiration: Mars exploration - Fetch mission



**Constraints:** Open source.  
Financially affordable.  
Simple to build.  
Programming in Scratch (10-14 ans) or Python (14-18 ans).

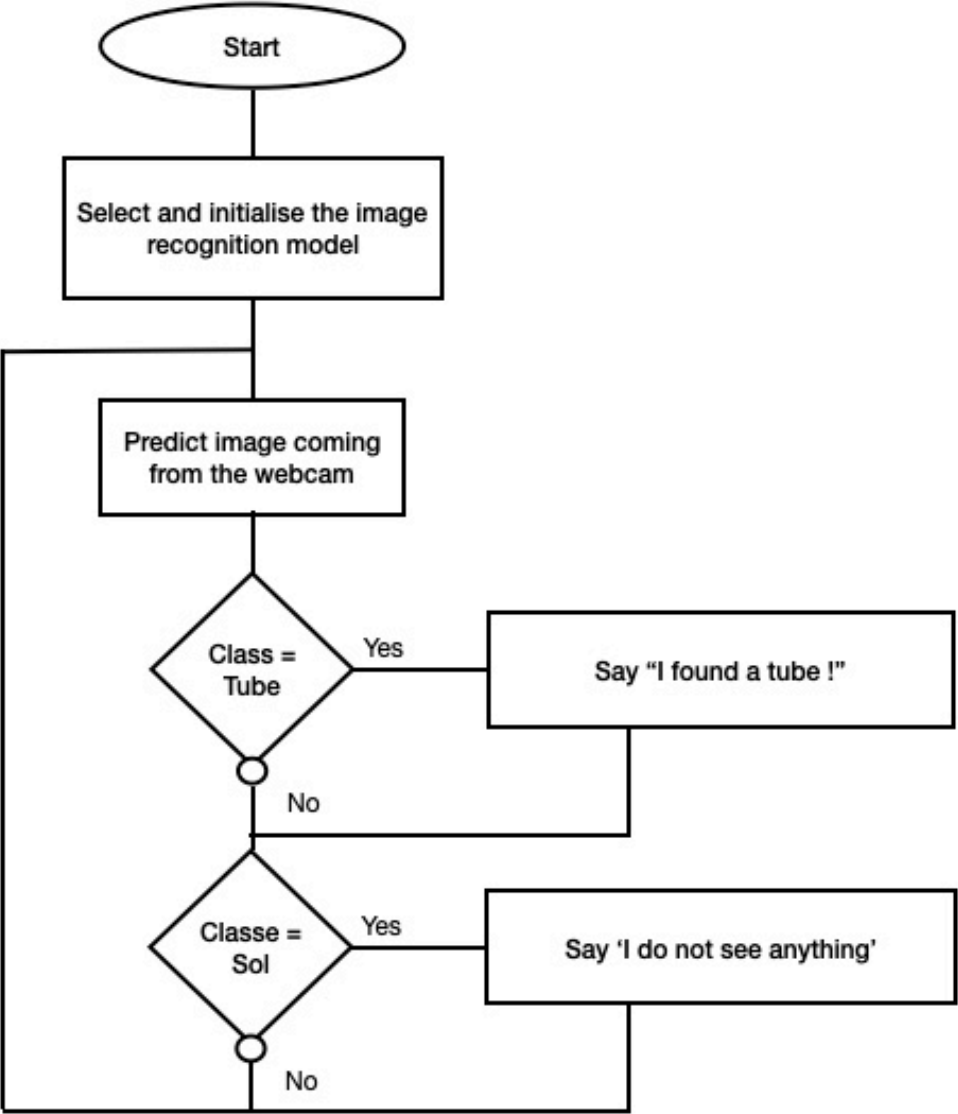
# Image recognition

## Train a model to detect tubes with the Teachable Machine

The screenshot displays the Teachable Machine web interface at <https://teachablemachine.withgoogle.com/train/image>. The interface is divided into several sections:

- Classes:** Two classes are defined: "Sol" and "Tube", each with 90 image samples. The "Tube" class is currently selected for training.
- Training:** A central panel shows the training progress. A "Model Trained" button is visible, and the "Advanced" settings are expanded.
- Preview:** A live preview window shows the model's output. The "Input" is set to "Webcam" and is turned "ON". The preview image shows a blue tube in a field of grass. The "Output" section shows the model's classification results: "Sol" with a low confidence bar and "Tube" with a high confidence bar at 100%.

# Image recognition with Scratch (Adacraft)



```
when green flag clicked clicked
  select and init the model which URL is https://teachablemachine.withgoogle.com/mode
  forever
    run detection on the webcam image
    if best detection class = Tube then
      say I found a tubel for 1 seconds
    if best detection class = Sol then
      say I do not see anything for 1 seconds
```

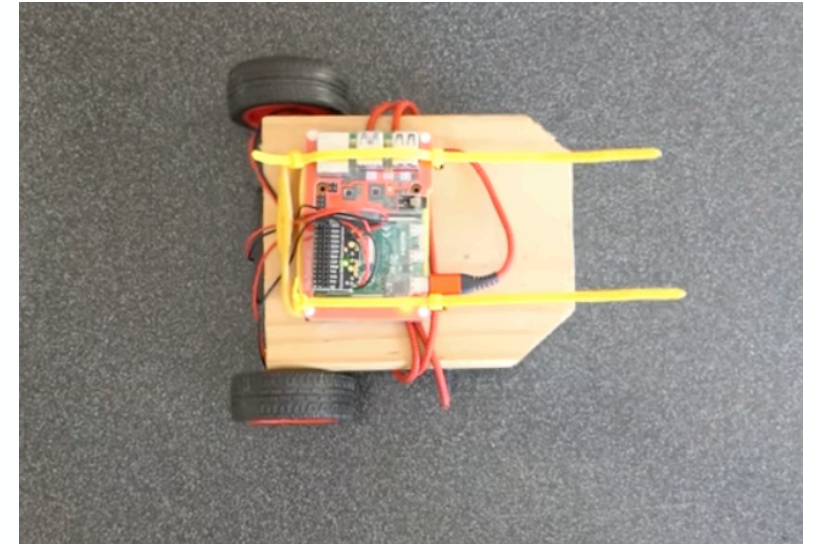
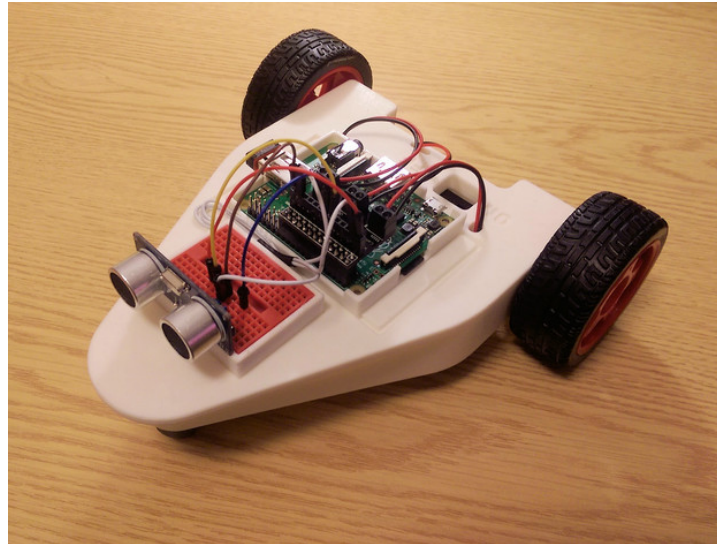
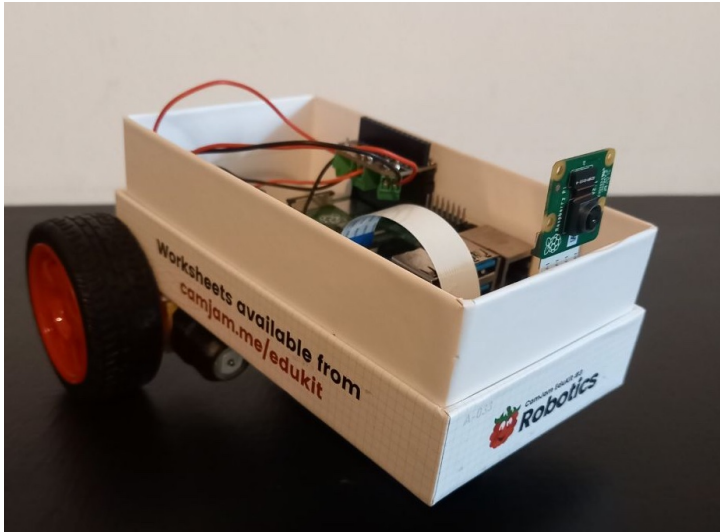
The Scratch code implements the logic from the flowchart. It starts with a 'when green flag clicked' event, followed by a 'select and init the model' block with the URL <https://teachablemachine.withgoogle.com/mode>. A 'forever' loop contains a 'run detection on the webcam image' block, followed by two 'if' blocks. The first 'if' block checks if the 'best detection class' is 'Tube', and if true, it says 'I found a tubel' for 1 second. The second 'if' block checks if the 'best detection class' is 'Sol', and if true, it says 'I do not see anything' for 1 second.

# Image recognition with Scratch (Adacraft)

The screenshot displays the Adacraft Scratch studio interface. The browser address bar shows <https://www.adacraft.org/studio/>. The top navigation bar includes 'adacraft', 'Fichier', 'Modifier', 'Projet\_Detection\_Mouvement', 'Voir la page du projet', and 'Feedback sur adacraft'. The left sidebar contains various tool categories: Contrôle, Capteurs, Opérateurs, Variables, Listes, Mes Blocs, Addons, Ada Vision, Musique, Style, and Détection vidéo. The main workspace shows a script starting with 'vidéo activée', followed by 'mettre la transparence vidéo sur 50'. A 'quand est cliqué' event triggers a 'sélectionner et initialiser le modèle dont l'URL est' block with the URL [https://teachablemachine.withgoogle.com/models/fi\\_pKOhf59/](https://teachablemachine.withgoogle.com/models/fi_pKOhf59/). This is followed by a 'répéter indéfiniment' loop containing two conditional blocks: 'si classe détectée = Tube alors dire J'ai détecté un tube ! pendant 1 secondes' and 'si classe détectée = Autre alors dire Je ne détecte rien... pendant 1 secondes'. The right panel shows a 'Rover' sprite on a scene with a speech bubble saying 'J'ai détecté un tube !'. The sprite's properties are set to x: 3, y: -85, Taille: 30, and Direction: 90.

# FetchBot: Hardware

- Raspberry 3 or 4, and camera (AstroPi Kit)
- Rover: CamJam EduKit (Around 20 euros)





# Training the rover to avoid obstacles and find tubes

The screenshot displays the Teachable Machine web interface. On the left, three categories are listed: 'Soil' with 21 image samples, 'Obstacle' with 20 image samples, and 'Tube' with 21 image samples. Each category has 'Webcam' and 'Upload' buttons. In the center, a 'Training' panel shows a 'Model Trained' button and an 'Advanced' dropdown. On the right, a 'Preview' panel shows a live webcam feed of a red tube on a reddish-brown soil surface. Below the feed, the 'Output' section shows three bars: 'Soil' (light orange), 'Obsta...' (light pink), and 'Tube' (purple) which is at 100%.



# Control the rover to avoid obstacles and find tubes

- If class is 'Soil' then move forward
- If class is 'Obstacle' then turn left
- If class is 'Tube' then say 'I found a tube!'

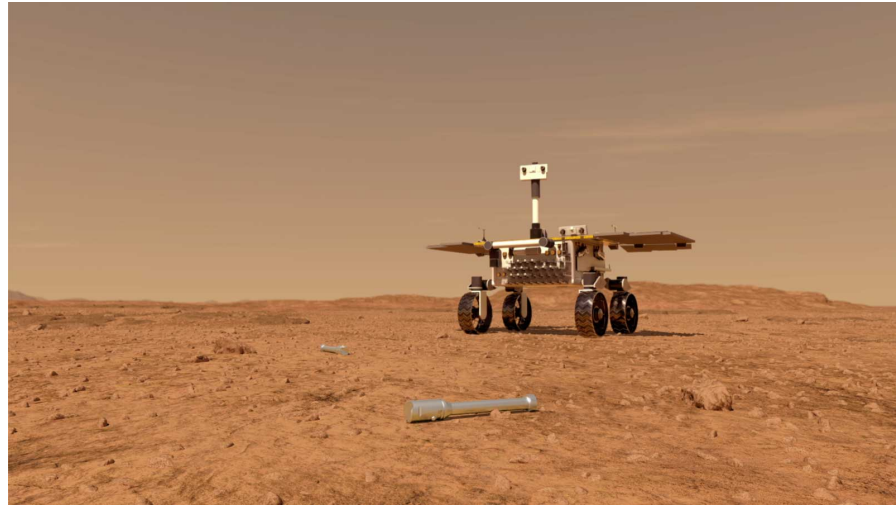


```
forever
  run detection on the webcam image
  if best detection class = Soil then
    say I see the soil
    Send command direct id Forward
  if best detection class = Obstacle then
    say I see an obstacle
    Send command direct id Left
  if best detection class = Tube then
    say I found a tube!
```

The image shows a Scratch script for controlling a rover. It starts with a 'forever' loop. Inside the loop, the first block is 'run detection on the webcam image'. This is followed by three conditional blocks: 1) 'if best detection class = Soil then', which contains a 'say I see the soil' block and a 'Send command direct id Forward' block. 2) 'if best detection class = Obstacle then', which contains a 'say I see an obstacle' block and a 'Send command direct id Left' block. 3) 'if best detection class = Tube then', which contains a 'say I found a tube!' block. The script ends with a return arrow at the bottom of the 'forever' loop.

# FetchBot: Demo

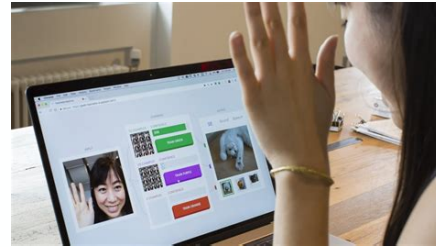




FetchBot project page :

<https://lascientotheque.github.io/fetchbot-fr>

# Summary and extensions



## Teachable Machine

10-14

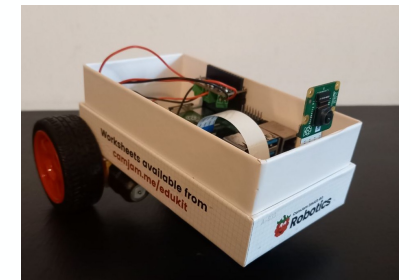
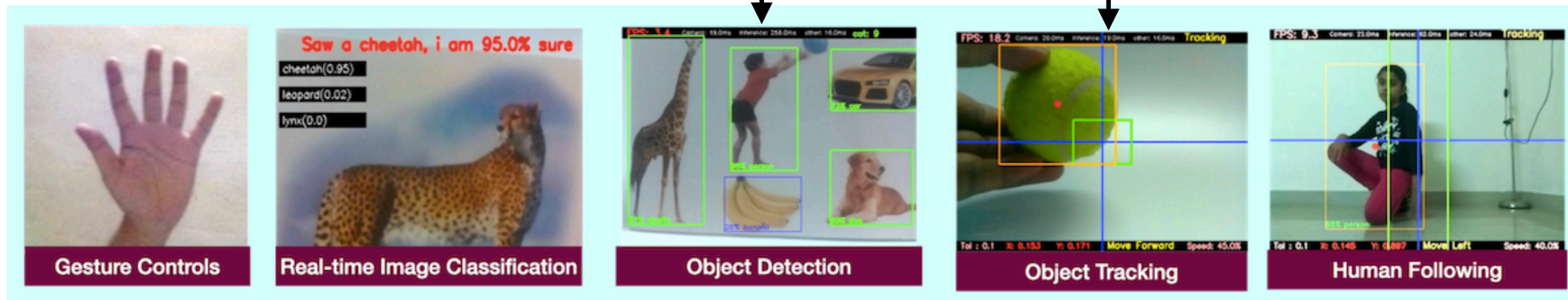


14-18

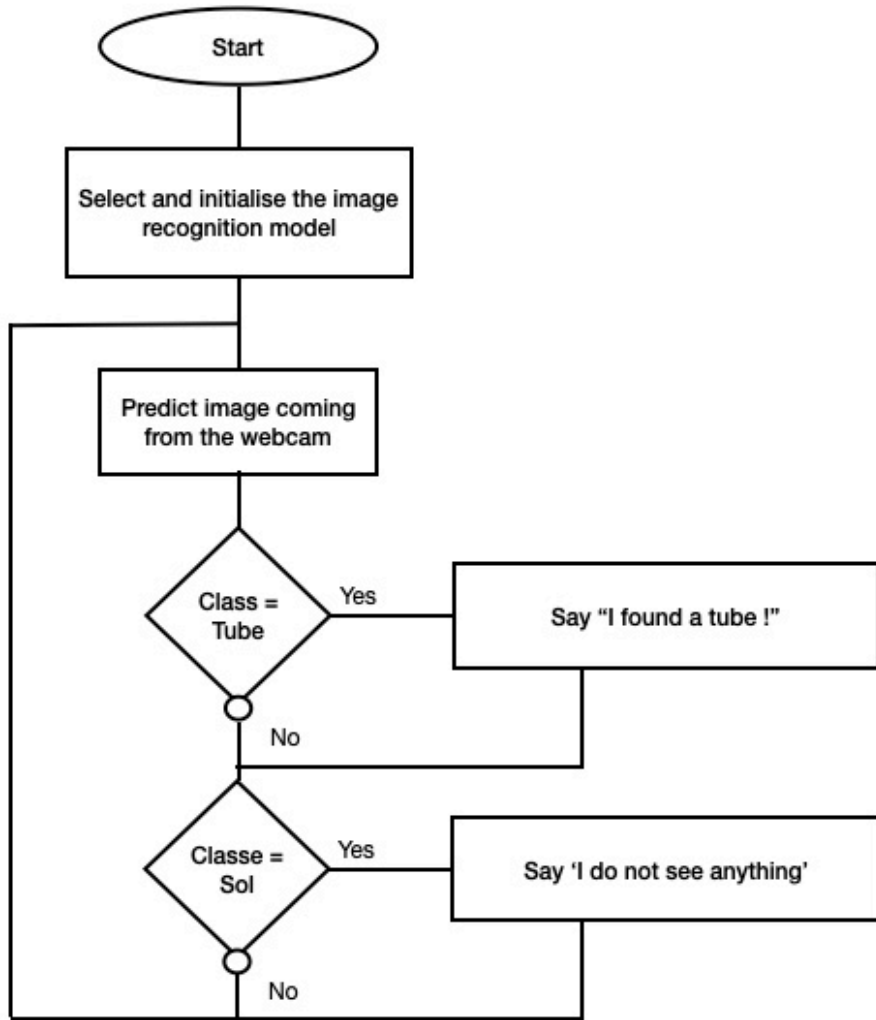
Export model

Implementation

Applications



# Image recognition with Python



```
import cv2
import myfunctions
```

```
# Intialisation de la caméra
camera_object = cv2.VideoCapture(0)
```

```
# Intialisation du modèle
interpreter = myfunctions.initialize_model(model_path='model_unquant.tflite')
```

```
# Répéter indéfiniment
while True:
```

```
    # Prendre une image de la caméra
    picture_rgb = myfunctions.take_picture(camera_object)
```

```
    # Prédire la classe de l'image
    prediction, probability = myfunctions.model_prediction(interpreter, picture_rgb)
```

```
    # Si la prédiction est la classe 0, alors la prédiction est sol
    if prediction == 0:
```

```
        print("Je vois le sol")
```

```
    # Si la prédiction est la classe 1, alors la prédiction est tube
    if prediction == 1:
```

```
        print("J'ai trouvé un tube!")
```