

IDEATION SHEET - EXPLORATION ACTIVITY

Mbot2 Navigator

AI exploration phase

Validating models through robotic activities

AI and Exploration Topics. What are we exploring?

Students explore how robots use computer vision and pre-trained AI models to interact with the physical world. Through coding with the mBot2 and Smart Camera, participants learn how visual perception translates into physical decisions the input, processing and output.

Materials Needed and to Create. What must be prepared before the activity?

An introductory presentation
mBot2 + bluetooth dongle
mBlock Smart Camera's
Laptops with mBlock 5 installed
Visual recognition targets (e.g. 3D printed)
Activity track (clean floor or a grid mat)



Exploration Steps and Description. What do participants actually do?

We start with a presentation on AI and computer vision, learning how machines rely on visual patterns rather than true understanding. Next, we connect to mBlock 5 and program basic mBot2 movements calibrated to the Smart Camera. By applying the Sense-Think-Act loop, students code the robot to make real-time decisions based on camera inputs, such as stopping at a red sign or turning at a specific marker.

Expected Output. What do participants create, produce or gain from the exploration?

Participants create a mBlock script using real time Smart Camera inputs. Through this they create an understanding of the sense think act loop.

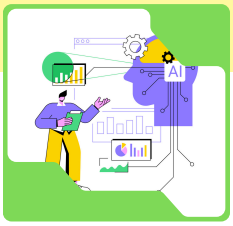
The Hook and the Playfulness. What makes this fun/exciting for young people?

The fun lies in the mBot2 that reacts to live environments. A self driving car is a concept thats relevant right now.

Success indicators. How do we know it worked? What shows participants learned?

Succes is clear when the mBot2 navigates automonously using its camera. And when participants can troubleshoot physical failures.

Co-funded by the European Union
CC BY SA



Appendix - Example of material

Appendix 1 | Introductory presentation

Content of the appendix

Part one: the Introductory presentation

How AI "Learns": Training vs. Traditional Code

- Rule-Based vs. Data-Driven: Traditional code uses fixed rules ("If X, do Y"). AI works differently: instead of hardcoding every rule, we feed it data and examples so it finds patterns on its own.
- The Training Process: Just like a child learning what an apple is by seeing hundreds of apples, machine learning algorithms process thousands of sample images to recognize visual features (edges, colors, ratios).
- The Model: The result of this learning is a trained model. The mBot2 Smart Camera carries a lightweight model that turns pixels into recognizable objects, like stop signs or arrows in real time.

From pixel to action: Input, Processing and output

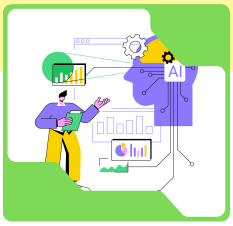
- Input: The camera registers the physical world as a matrix of pixels and color values and recognizes a pattern ("Red patch detected at position X, Y").
- Processing: The mBot2 processor receives this input and executes the programmed code using if-then logic ("If red is detected, trigger command: stop").
- Output: The motors execute the physical action by coming to a stop, or the screen displays a stop icon.

AI makes mistakes (Chihuahua vs Muffin paradox)



Source: Zack, Karen [@teenybiscuit].
"chihuahua or muffin ?" X, 10 mrt. 2016, x.com

Co-funded by the European Union
CC BY SA



Appendix - Example of material

Appendix 1	Introductory presentation

Content of the appendix

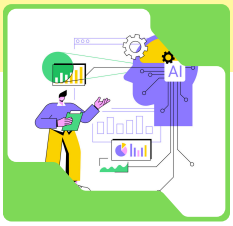
Chihuahua VS Muffin Paradox

- Why Does AI Make Mistakes? AI doesn't understand reality the way humans do. It doesn't grasp the concept of an actual dog. it analyzes numbers, color gradients, pixels, and contrast points.
- The Famous Example: Chihuahua vs. Blueberry Muffin Show students the classic grid comparing Chihuahuas and blueberry muffins
- We instantly spot the difference between a living animal and muffins
- To a Computer Vision algorithm, three dark spots (eyes and nose) on a beige background look mathematically very similar to three blueberries baked into a scoop of muffin batter.

The Link to the mBot2 Smart Camera:

- If the mBot2 camera misreads a sign the algorithm sees matching pixel patterns in the wrong context.
- How this affects our computing chain:
  - Flawed Input: Poor lighting or an odd angle distorts the image data.
  - Confused Processing: The AI's pattern-matching score slips (e.g., "51% confidence this is a stop sign").
  - Incorrect Output: The mBot2 slams on the brakes unexpectedly or drives right through a stop marker.

Co-funded by the European Union
CC BY SA



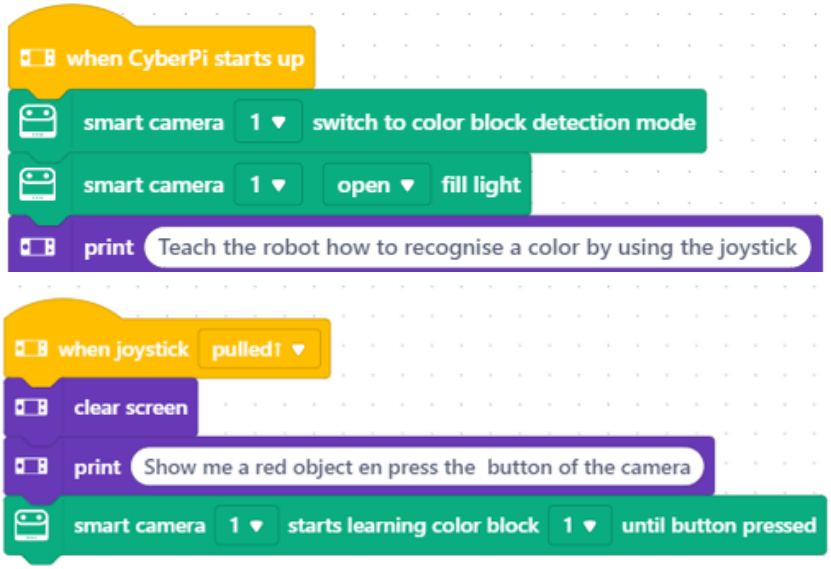
Appendix - Example of material

Appendix 1	Coding with mBot

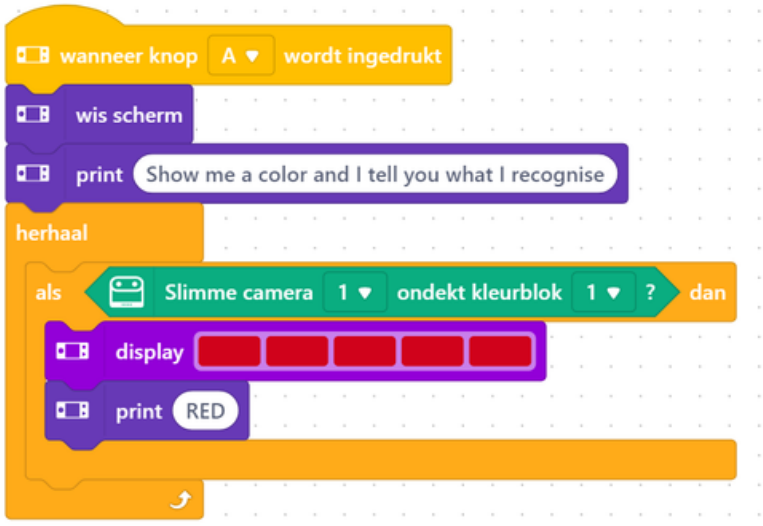
Content of the appendix

Machine Learning happens inside the Smart Camera.
Behind the scenes, the camera uses a trained Machine Learning model to analyse what it sees and recognise different objects or images. The model produces a recognition result and can also provide a confidence score for that prediction.

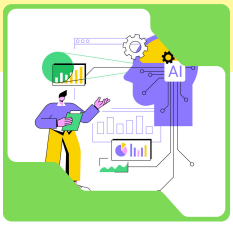
Lets program!
First we program what we need to do to teach the smart camera to learn a color for example red:



Lets add code to verify if the robot recognises the color red:



CC BY SA



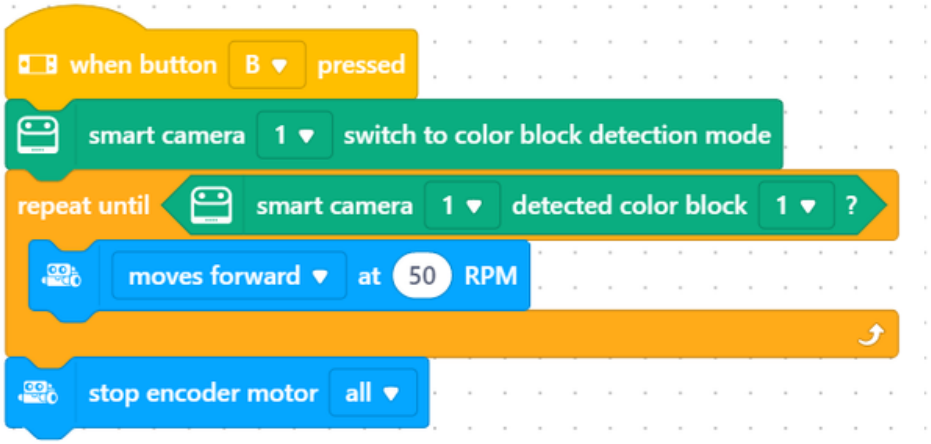
Appendix - Example of material

Appendix 1	Coding with mBot

Content of the appendix

The Smart camera has now learned to recognise our color.
Let's now do something with this information.

We are going to connect the camera's prediction to the robot's movement.
The robot will go through a sense → think → act loop
Recognise the color red → Make a decision → take action
Let's make the robot drive and use the red object as a stop sign.
We already used the Joystick and button A in previous commands. So we now use button B.



The same idea can be used in many different ways. We can change the program to react to different colors and make the robot react differently to each colour. For example, it could stop at red, turn at blue, and keep driving at green. By changing the code, we can teach the robot different rules for different colours.

Co-funded by the European Union
CC BY SA