

Challenges and opportunities of the new era
Manual for teaching older students

ARTIE



ARTIE

Challenges and opportunities of the new era: A Handbook for Teachers

Authors

Željko Krnjajić
Janko Radigović

Publisher

Hrvatski robotički savez, Croatia
“Artificial Intelligence in Education - challenges and opportunities of the new era: development of curriculum, guides for teachers and e-courses for students”, project number 2020-1-HR01-KA201-077800 under Erasmus+ Programme.

Consultants/ Reviewers

Katarzyna Garbacik
Andrzej Garbacik
Bogusław Klimczuk
Ivana Ružić
Ana Pina
Christina Eirini Karvouna

Graphic Design & Illustrations

Christina Eirini Karvouna

Translators

Jura Cmrečak (Croatian)
Bogusława Denys (Polish)
Ana Pina (Portuguese)
Christina Eirini Karvouna (Dutch)



3	Artificial intelligence in robotics
11	Face detection and recognition in Scratch
19	Programming face detection in Scratch
26	Programming facial recognition in Scratch
41	Object detection and classification for beginners
49	Object detection programming in Scratch
57	Project with object classification
63	Speech recognition and generation for beginners in Scratch
68	Recognition programming of speech in Scratch
73	Programming speech generation in Scratch
77	Voice control project object
83	Introduction to hardware - microcontroller, camera and engine controller
93	Interfacing the microcontroller with camera and computer, start of work
100	Assembling the robot
119	Programming of robot movements
128	Robot programming - object tracking
140	Programing a robot - line following



Teaching scenario 1

Duration: 90 min

Topics

artificial intelligence in robotics, artificial intelligence, robotics

Aims

To get familiar with and understand AI in robotics

Outcomes

Understanding how AI is used in robotics

AI in robotics

The purpose of this lesson is to learn what artificial intelligence is in robotics and how it can be used in our daily lives.

The teacher announces the topic and starts the discussion:

Could AI in robotics change the future?

What is artificial intelligence in robotics?

Do we use robots with AI already in our everyday life? How? Where?

Announcement of the goal of the lesson

Explain what artificial intelligence is and discuss its current and future applications in robotics



<http://erasmus-artie.eu>



MAIN PART

Topics for discussion:

What is a robot?

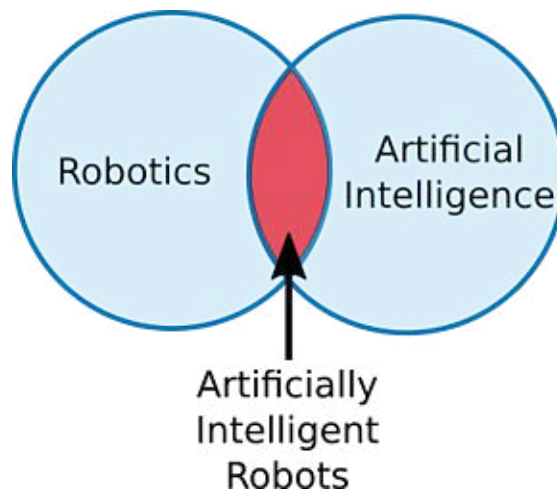
Does every robot operate with AI?

Do you know some examples of AI in robotics?

Could robots with AI replace humans?

The first thing to clarify is that robotics and artificial intelligence are not the same things at all. In fact, the two fields are almost entirely separate.

A Venn diagram of the two fields would look like this:



There is a small area where the two fields overlap: Artificially Intelligent & Robots. It is within this overlap that people sometimes confuse the two concepts.

What is Artificial Intelligence?

Artificial intelligence (AI) is a branch of computer science. It involves developing computer programs to complete tasks that would otherwise require human intelligence. AI algorithms can tackle learning, perception, problem-solving, language-understanding and/or logical reasoning. According to the father of Artificial Intelligence, John McCarthy, it is also “The science and engineering of making intelligent machines, especially intelligent computer programs”.



Simply put, Artificial Intelligence is a way of **making a computer, a computer-controlled robot, or a software think intelligently**, in the similar manner the intelligent humans think.

Goals of AI

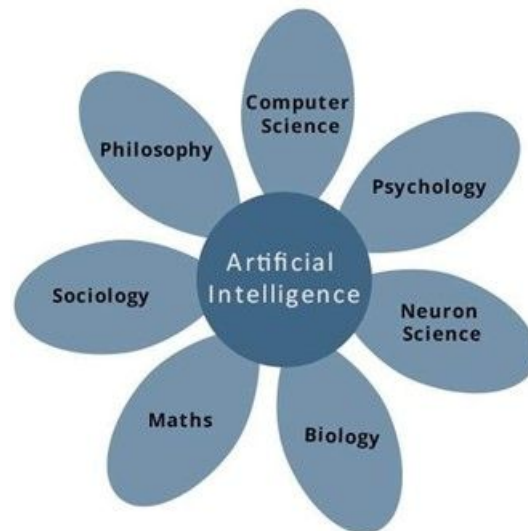
to create expert systems – these systems exhibit intelligent behavior, learn, demonstrate, explain, and advise its users.

to implement human intelligence in machines – such systems would understand, think, learn, and behave like humans.

What Contributes to AI?

Artificial intelligence is a science and technology based on disciplines such as Computer Science, Biology, Psychology, Linguistics, Mathematics, and Engineering. A major thrust of AI is in the development of computer functions associated with human intelligence, such as reasoning, learning, and problem solving.

Out of the following areas, one or multiple areas can contribute to building an intelligent system.



What is Robotics?

Robotics is a branch of AI, which is composed of Electrical Engineering, Mechanical Engineering, and Computer Science for designing, construction, and application of robots.

What are Robots?

Robots are artificial agents acting in a real world environment.

They are programmable machines that are usually able to carry out a series of actions autonomously, or semi-autonomously.



Objective

Robots are developed to have the ability to manipulate objects by perceiving, picking, moving, modifying the physical properties of object, destroying it, or to free manpower from doing repetitive tasks without getting bored, distracted, or exhausted.

Aspects of Robotics

The robots have **mechanical construction**, form, or shape designed to accomplish a particular task.

They have **electrical components** which power and control the machinery.

They contain some level of **computer program** that determines what, when and how a robot does something.

Artificial intelligence in robotics

AI in robotics helps robots perform the crucial tasks with a human-like vision to detect or recognize various objects. Robots are developed through machine learning and training and huge number of datasets is used to train the computer vision model, so that robots can recognize various objects, carry out the actions accordingly and accomplish desired outcomes. Computer vision is simply the process of perceiving the images and videos available in digital formats. The AI in robotics not only helps to teach the model to perform certain tasks, but it also makes machines more intelligent and therefore able to act in different scenarios.

Here are some examples of most advanced Humanoid, Industrial and Service robots that are changing the future with the help of Artificial Intelligence.

Sophia

Hanson Robotics' most advanced human-like robot, Sophia, personifies our dreams for the future of AI. As a unique combination of science, engineering, and artistry, Sophia is simultaneously a human-crafted science fiction character depicting the future of AI and robotics, and a platform for advanced robotics and AI research.

The character of Sophia captures the imagination of global audiences. She is the world's first robot citizen and the first robot Innovation Ambassador for the United Nations Development Programme. Sophia is now a household name, with appearances on the Tonight Show and Good Morning Britain, in addition to speaking at hundreds of conferences around the world.

Meet Sophia: <https://www.youtube.com/watch?v=BhU9hOo5Cuc>





Digit

Digit is envisioned to help take care of people in their homes, assist with disaster response, and deliver packages to front doors. With its nimble limbs and a torso packed with sensors, Digit can navigate complex environments and carry out tasks such as package delivery. In May 2019 Ford Motor Company and Agility announced a partnership to develop a last-mile logistics solution that combines Ford's autonomous vehicle technology and Agility's Digit.



Pepper

Pepper is the world's first social humanoid robot that is able to recognize faces and basic human emotions. Pepper has been adopted by over 2000 companies around the world. Perfect in retail and finance industries, Pepper has numerous functionalities including increasing store traffic by attracting the attention of shoppers, creating memorable in-store experiences, stimulating purchase and retain customers. Pepper can also gather comprehensive data to enrich the customer base and generate shopper insights.



<http://erasmus-artie.eu>

Atlas

Atlas is the world's most dynamic humanoid robot built by BostonDynamics, a company that was previously owned by Google and now by SoftBank. Atlas is becoming more sophisticated year by year, thanks to its state-of-the-art hardware and algorithm that allows it to quickly understand instructions. With its 28 hydraulic joints, 4.9 feet in height and 176 pounds in weight, the robot can perform both impressive and terrifying acts including navigating uneven terrain, jumping around a parkour course, and doing somersaults. All these activities demonstrate human-level agility so the robot can be perfect for search and rescue operations and performing human tasks in environments where humans could not survive.





Spot

Spot is a robot dog designed for industrial uses such as carrying goods through a warehouse and inspecting a remote site with an unfavorable environment for human operators. It can run at 5.2 feet per second, has 360-degree cameras, and can operate in temperatures ranging from 4 to 113 Fahrenheit. With its API and flexible payload interface, the robot can be easily customized for desired tasks. Spot is also manufactured by BostonDynamics and is now being leased to eligible companies.



f

HRP-5P

HRP-5P is an advanced humanoid robot designed to operate autonomously and carry out heavy labor in hazardous environments. It is equipped with environmental sensors and object recognition, full-body motion planning and control, and task description and execution management. HRP-5P is based on more than 20 years of humanoid research at AIST. In those 20 years, the institute has created 4 other robots which are the predecessor of HRP-5P.



<http://erasmus-artie.eu>

Surena IV

Surena IV is the fourth generation of Surena humanoid robot series developed by the University of Tehran in Iran. With a height of 5.6 feet and a weight of 154 pounds, this robot is able to walk at a speed of 0.43 miles per hour. Its custom force sensors at the bottom of its feet help the robot step over uneven surfaces by adjusting the angle and position of each foot.





Aquanaut

Aquanaut is an advanced unmanned underwater transformer that can transform itself from a nimble long-distance submarine into a half-humanoid robot capable of carrying out complex underwater manipulation tasks. Designed by Houston Mechatronics Inc, Aquanaut can inspect subsea oil and gas infrastructure, operate valves, and use subsea tools with just a few mouse clicks. Operating completely untethered and without support ships, Aquanaut can travel more than 124 miles in submarine mode, has a max speed of 7 knots and a maximum operational depth of 984 feet.



f

Stuntronic robot

A Stuntronic robot is an animatronic stunt double designed to entertain the crowds at Disney theme parks and resorts. With its onboard sophisticated sensors, it can make its own real-time decisions—all that while flying at 60 feet up in the air. It knows when to tuck its knees to perform a somersault, when to pull its arms to twist, and even when to slow down its spin to make sure it makes a perfect landing.



<http://erasmus-artie.eu>

Handle

Handle is another robot from Boston Dynamics. With its deep-learning vision software, this robot can identify and locate boxes, unload trucks, palletize, and depalletize at the push of a button. Its mobility enables it to operate in multiple work-cells, moving through facilities along with the flow of goods. It can pick up to 360 boxes/hr.





Show students video (optionally) and discuss;
<https://www.youtube.com/watch?v=Jky9I1ihAkg>

Today we are using AI in robotics in healthcare, agriculture, automotive industry, at warehouses, at supply chain ...etc. Later on, we are going to build our own robot with AI and train him to do a face detection and recognition, object detection and speech recognition.

Here is a mobile robot with camera and AI capabilities developed exclusively for this project.

We will show you how to make it and use it for:

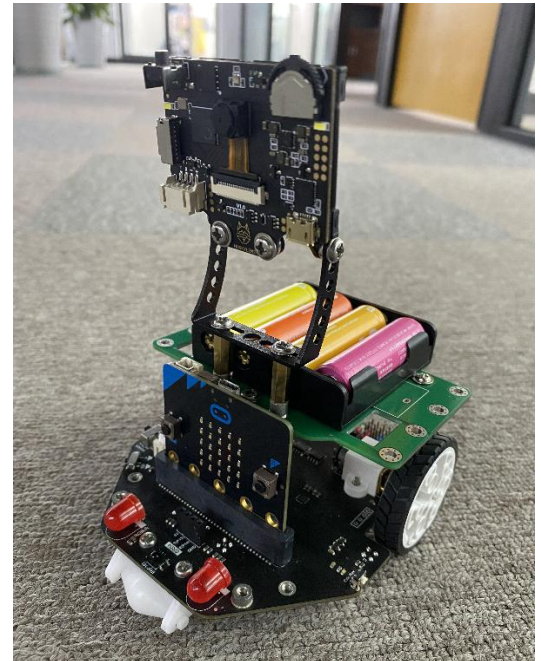
- Face detection
- Face recognition
- Face tracking
- Object detection
- Object tracking
- Line following

Artificial Intelligence is finally here and most of us are already actively using it in our day-to-day life (even without knowing it). Future generations need to understand how to use AI first of all! Only then can they use it to facilitate learning and solve real-world problems.

Artificial Intelligence (AI) and Robotics are strongly connected today.

AI in robotics is used in everyday life more and more and has been instrumental in the various domains such as industries, military, medicine, exploration, entertainment.

Remember, AI is most probably the most powerful technology ever invented by man. It can be used for both good and bad things. In the end, it's up to us how to use it.



CONCLUSION

Artificial intelligence is the way a computer, a computer-controlled robot or program thinks intelligently, in a similar way that intelligent people think



Learning scenario 2

Duration: 90 min

Topics

detection and facial recognition

Aims

To learn about face detection and recognition for beginners in Scratch

Outcomes

Becoming familiar with face detection and recognition with the help of simple examples in Scratch,
Understanding the difference between Face detection and Face recognition

Face detection and recognition for beginners in Scratch

Students should be asked to define what face recognition and face detection are.
Give them an opportunity to describe the difference.

Terms face detection and face recognition are sometimes used interchangeably, but there are some key differences. To help clear things up, let's look at the term face detection and how it differs from the term face recognition.

Announcement of the goal of the lesson:

Introduction to face detection and recognition for beginners through examples of different applications.



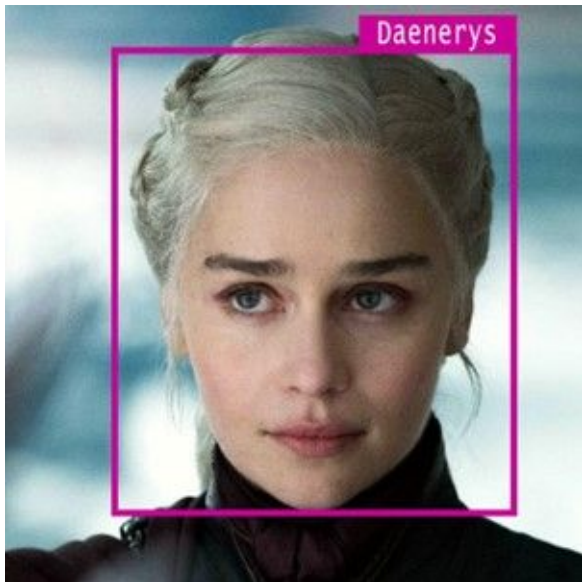
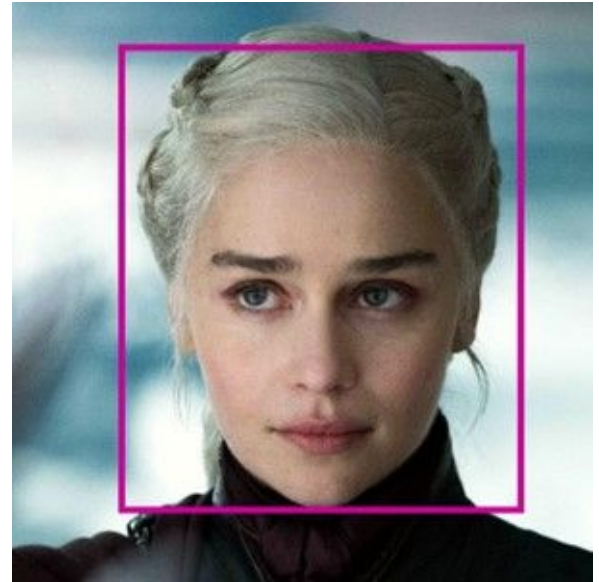
<http://erasmus-artie.eu>

11



MAIN PART

Face detection is a process that determines the presence of face(s) in a still image or in a video clip. For example, such a function is available in most smartphone camera software. But the face detection module does not determine whose face is in the frame.



Face Detection does not memorize or save facial features. If the software detects a face of some particular person in the frame and later finds the same face on another image, it will not determine that the face belongs to the same person; it will just detect the presence of a face in the frames. The software can provide data regarding age and sex of a person on each frame, but no more than that. Face Detection software cannot recognize particular persons.

In contrast, Face Recognition relates to identifying and recognizing persons.

The purpose of software-based face recognition is to perform identification of persons appearing in a still image or a video clip by comparing it to a database. To ensure successful identification, the corresponding faces must first be entered into the database. The software determines the unique features of a face, saves them and uses them for subsequent identification. Later, during the identification process, the software will compare the unique features and identify a face of a particular person in case these features match.



<http://erasmus-artie.eu>

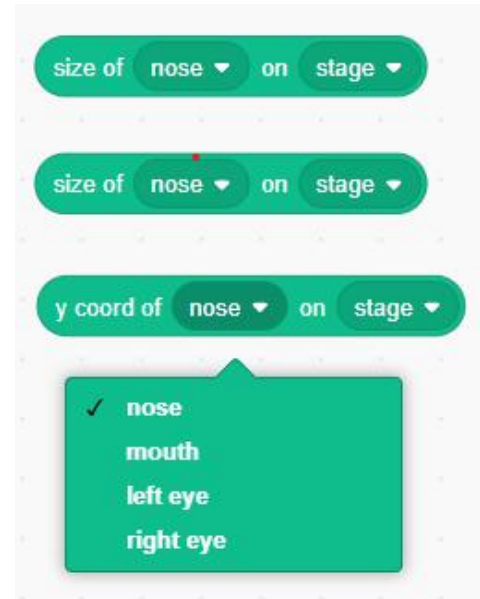


Scratch/Scratch based and other applications to use

Scratch (ML4KIDS)

<https://machinelearningforkids.co.uk/scratch3/>

Face detection extension with 3 reporter type blocks is available. In case that you're using a web camera as a source, combine it with Video sensing extension to turn camera video on/off and set transparency.

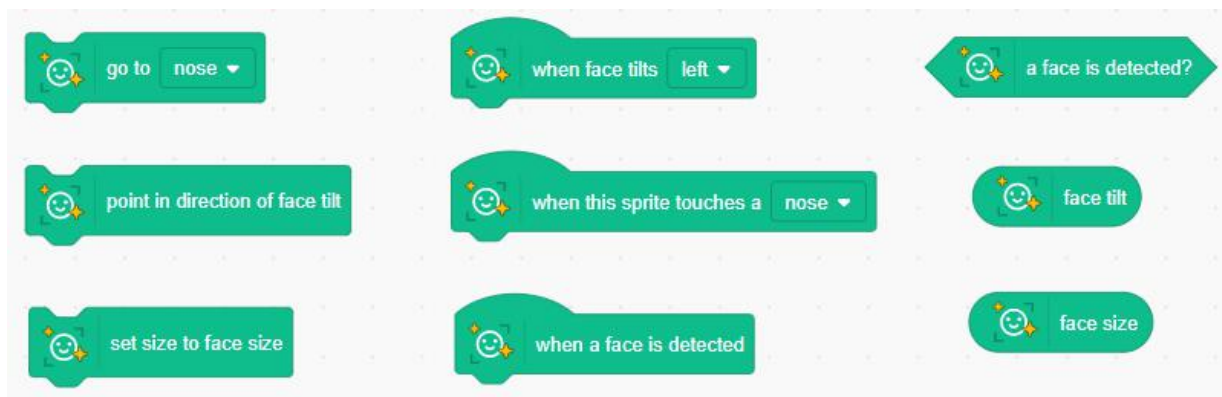


Scratch (MIT) - <https://lab.scratch.mit.edu/face/>

Click on "Try it out" and you will have 9 blocks for face recognition handling.



<http://erasmus-artie.eu>





Scratch (MITMEDIALAB) - <https://mitmedialab.github.io/prg-extension-boilerplate/create/>
 Load Face sensing extension and you will have 9 blocks for face detection handling. Some blocks are used for facial expression and feeling recognition. You can also use Teachable machine extension in combination with Google Teachable machine.

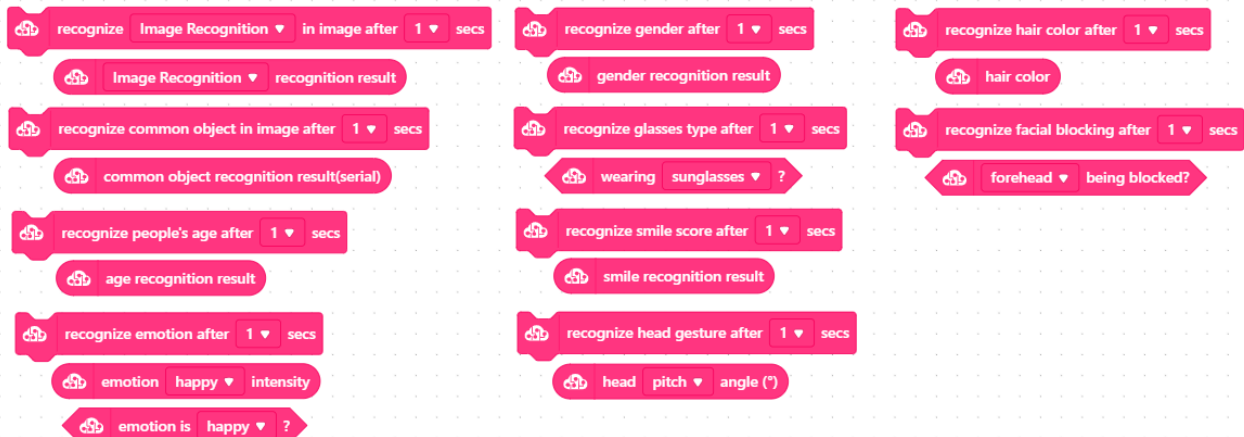


http://erasmus-artie.eu

Makeblock (mBlock) - <https://ide.mblock.cc/>

Load Cognitive services and Video Sensing extensions and you will have variety of blocks. There is no specific block for face detection, but you can detect a person inside a “recognize” block. But plenty of blocks are there to handle emotion, age, gender, smile, hair color, glasses and even a situation when you cover a part of your face.

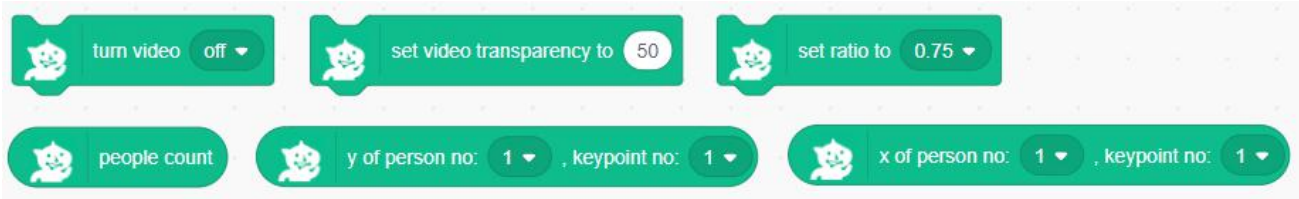
Makeblock also provides Teachable machine extension (not to be confused with Google’s) where you can train up to 3 classes and use it for facial recognition or object detection.





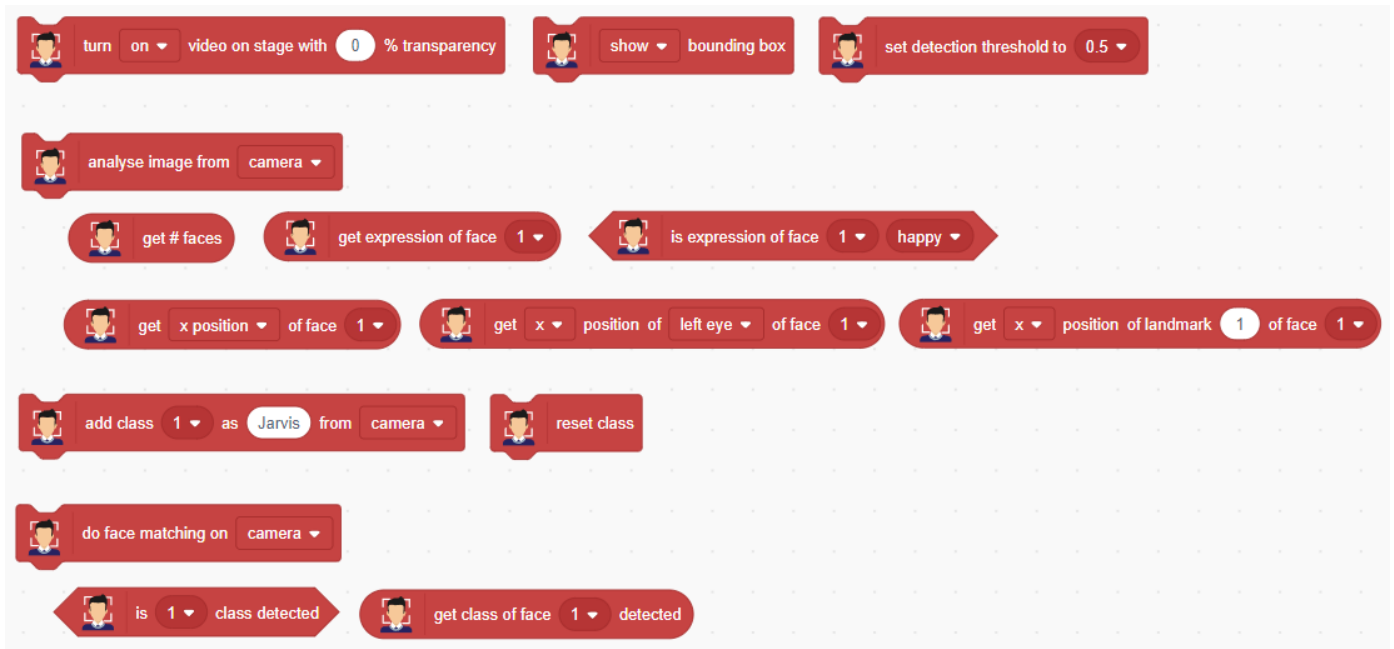
Stretch3 (github.io) - <https://stretch3.github.io/>

Load **Facemesh2Scratch** extension to use 3 blocks for face detection (there are additional 3 blocks for handling video). Main feature is multiple face detection capability so you can detect more than one person on your camera stream.



The last one to use is **PictoBlox**, a desktop type application which must be installed first from <https://thetempedia.com/product/pictoblox/download-pictoblox/> (427 Mb)

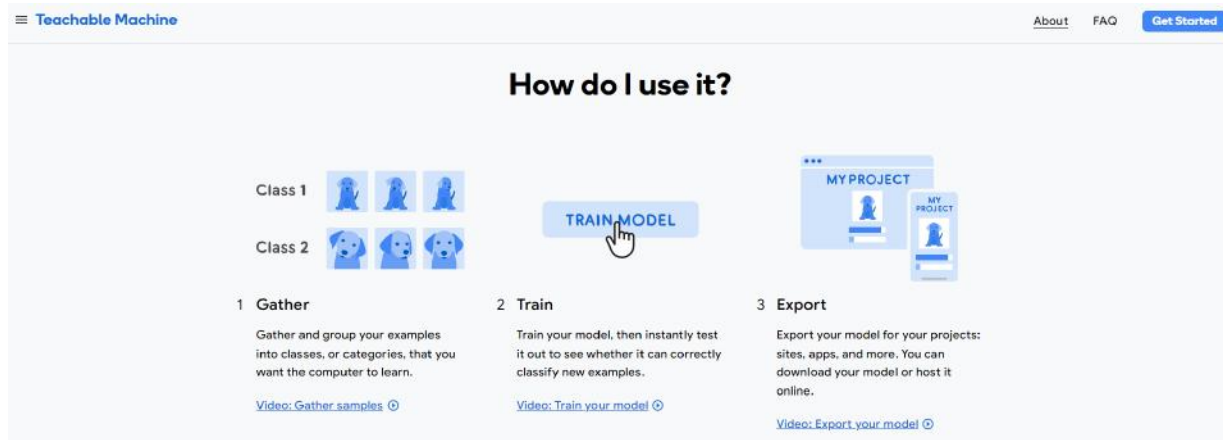
After installation use Face Detection extension and there you will see a pure treasure - you can detect multiple faces and their facial expressions. Also, there is a feature to train classes which leads us directly to facial recognition.





Teachable machine (Google) - <https://teachablemachine.withgoogle.com/>

This application is used to train your model and use it for face recognition in combination with Teachable machine extension available in Scratch (MITMEDIALAB)



We have seen three different applications but also a very similar one for face detection and recognition.

Face Detection differs from Facial Recognition (the terms should not be used interchangeably) in that Face Detection involves only the detection of a face within a digital image or video. It simply means that the face detection system can identify that there is a human face present in an image or a video – it cannot identify the person. Face detection is a component of Facial Recognition systems – the first stage of facial recognition is detecting the presence of a human face in the first place. Face detection can also be used in cameras to help with auto-focus – you have probably noticed that on some digital cameras and phones, a small box will appear around the faces of people detected within the image, allowing the camera to prioritise focus on those faces. Identifying the presence of a human face is done using formulas and algorithms.

Typically, the first thing that a face detection system will look for will be the eyes as these are one of the easiest features to identify. Then it might also search for the presence of mouth, eyebrows, nose, and nostrils. Face detection is an important part of the facial recognition process, however, from a security perspective, there is no independent benefit of a face detection system – it simply recognises a face is present but has no idea about the identity of that face. Facial recognition is playing a vital role across a huge range of industries, especially in border control and law enforcement. Accurately identifying individuals is helping to improve security and safety at airports and in towns and cities around the world, and this can only be done using market-leading facial recognition systems.

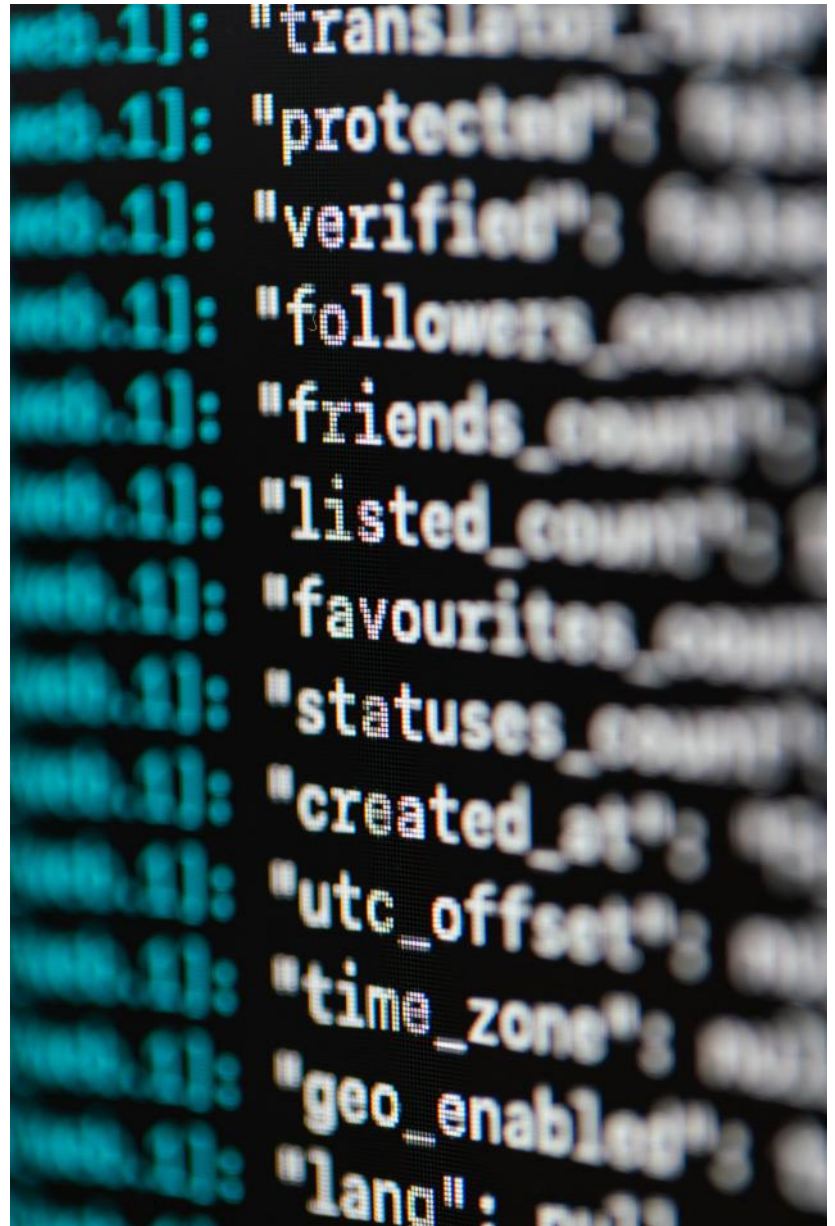
f

<http://erasmus-artie.eu>

17



As we can see, Facial Recognition technology is changing the world in which we live, and it feels like we are only just scratching the surface in terms of potential applications of facial recognition software. While uses for facial recognition may seem endless, we need to ensure that this technology is used appropriately and responsibly.



CONCLUSION

Face detection is a process that determines the presence of a face in an image or video clip while facial recognition refers to the identification and recognition of persons.



Duration : 90 min

Topics

Programing face detection
in Scratch

Aims

To learn to program face
detection with uploaded
examples

Outcomes

Knowing how to write a
program for face detection
using Scratch



Learning scenario 3

Programing face detection in Scratch

How to program face detection in Scratch?

We will review what we learned about face detection in the last class.

Ask your students about their experiences with face detection. Before you lead them in, ask them if they know how to make a face detection program.

The teacher introduces students to programing face detection in Scratch.

Let's walk through a few simple examples of face detection programing in Scratch and Scratch based application. You've probably noticed that some applications on your smartphone draw a rectangle around the face as a result of face detection. It is also possible to do that in Scratch.

Announcement of the goal of the lesson:

Understanding face detection software and its usage by going through specific examples.



<http://erasmus-artie.eu>

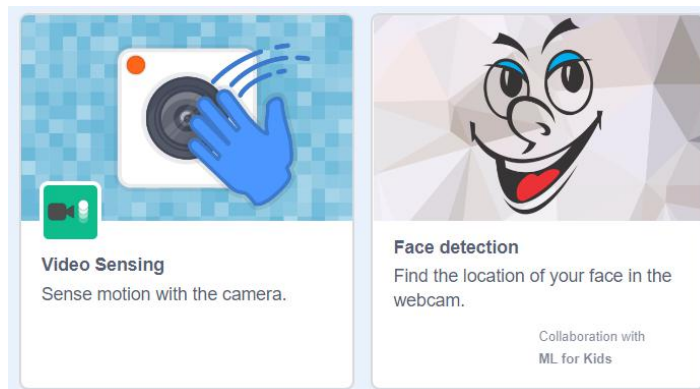


MAIN PART

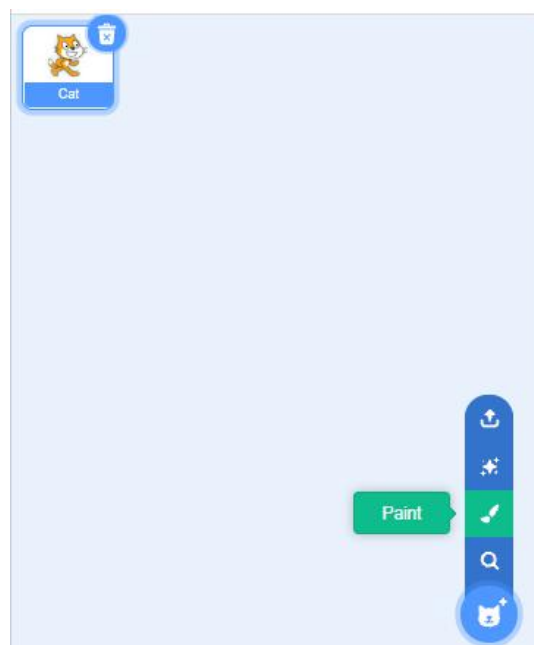
First project - SCRATCH (ML4KIDS):

Step 1: Open your Chrome web browser and go to:
<https://machinelearningforkids.co.uk/scratch3/>

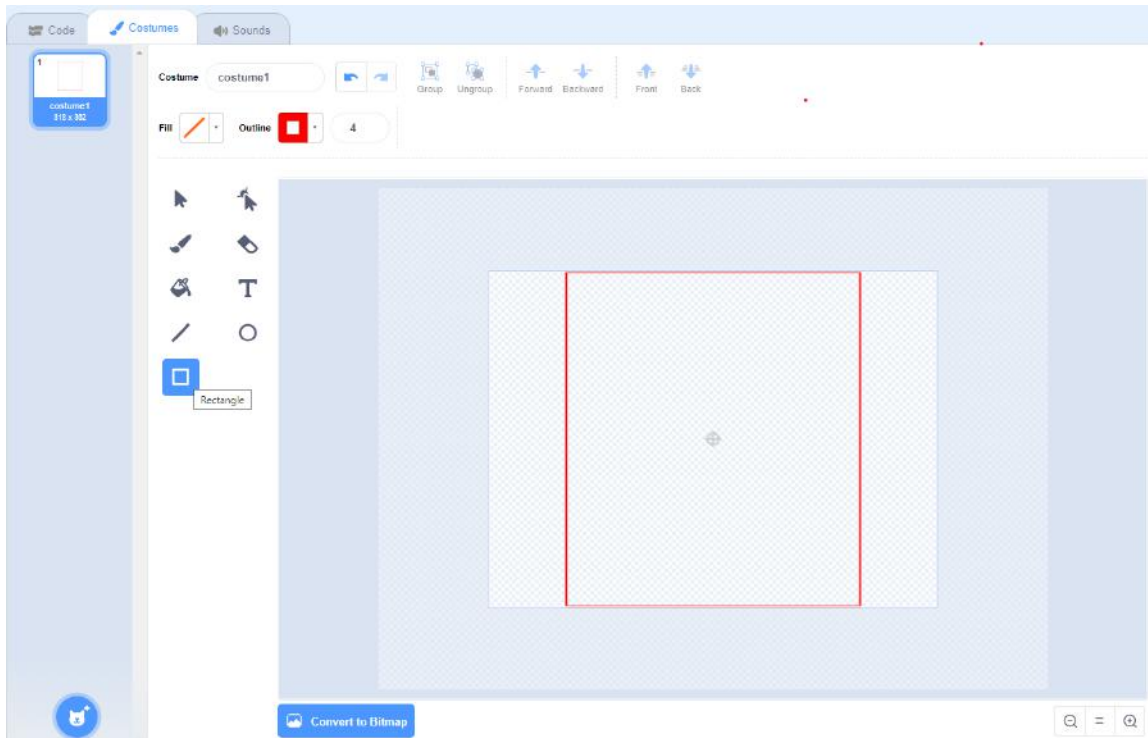
Step 2: Load Video Sensing and face detection extensions and plug in your web camera (if you don't have a built in one)



Step 3: Delete Cat sprite by clicking on trashcan icon (upper right corner) and choose the option to Paint a new sprite

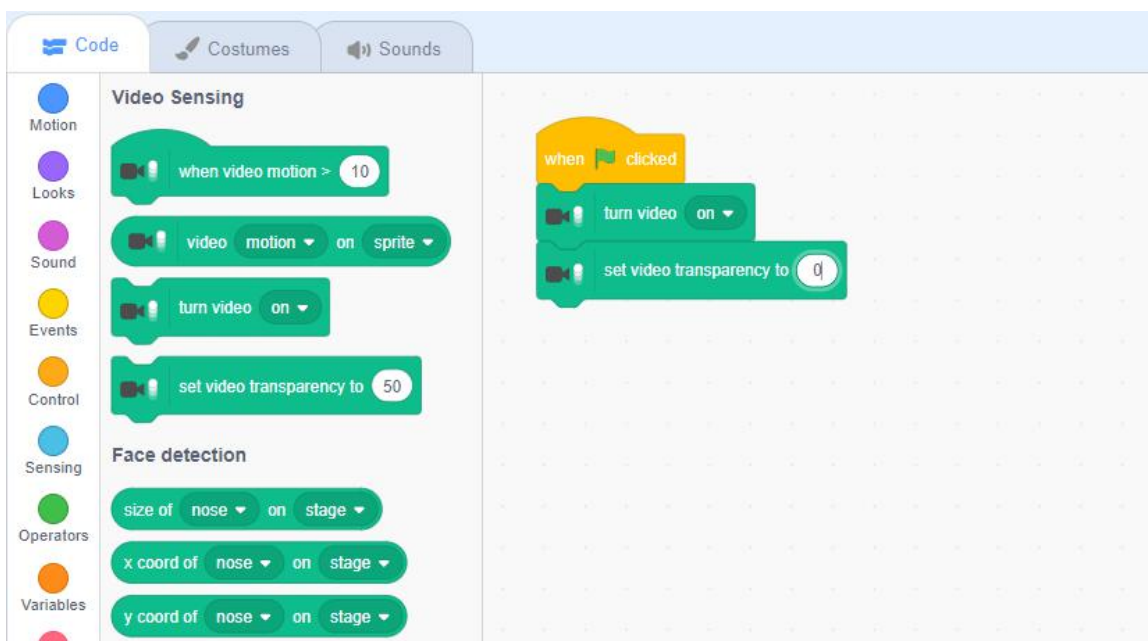


Step 4: Draw a rectangle (it will be used as a bounding box) with no fill and set the outline to red (4) as in picture below



f

Step 5: Switch to Code tab and start programming. First, we turn on the video and set the transparency to 0 (non-transparent).

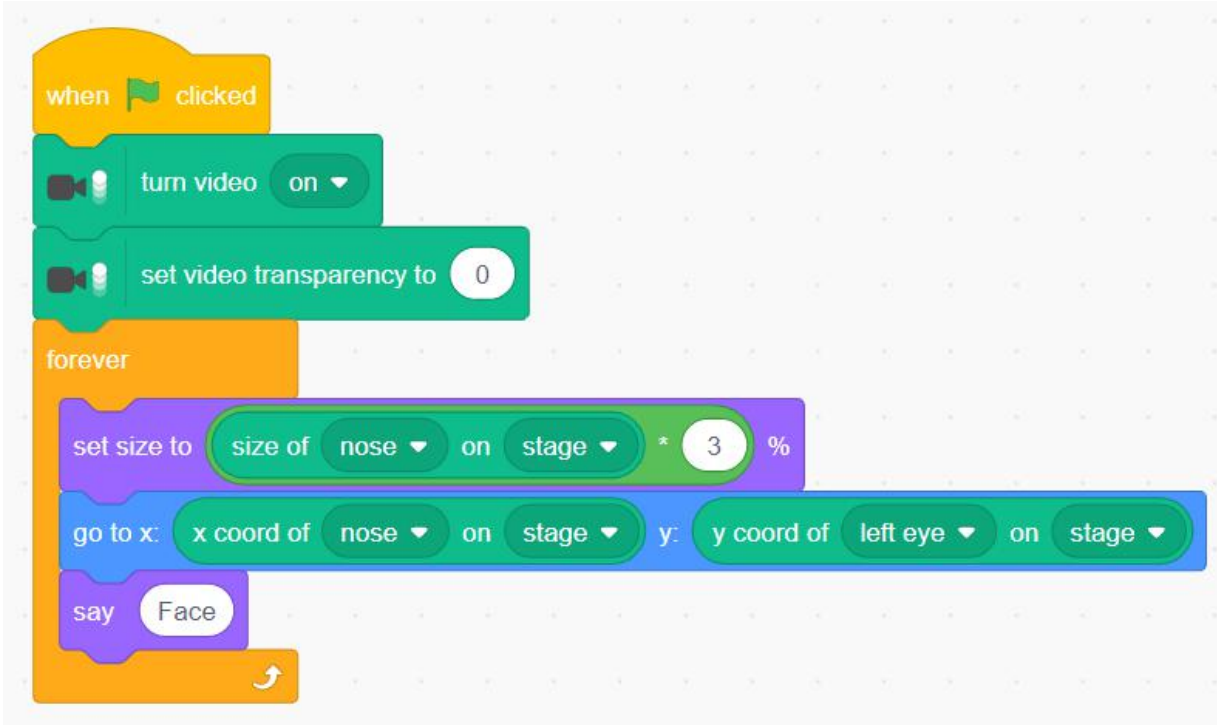


<http://erasmus-artie.eu>

21



Step 6: Next, there is a loop (forever) with 3 blocks in it to set the size of a rectangle (nose size multiplied by 3) and its x and y coordinates. You may want to adjust the value of a multiplier from 4 to some other number, especially if you switch to a bigger stage (1.5 is better).



Second project - face detection combined with augmented reality

Step 1: Open Scratch at <https://machinelearningforkids.co.uk/scratch3/>

Step 2: Add extension "Face detection"

Step 3: Add extension "Video sensing" and plug in your web camera (if you don't have built in one)

Step 4: Delete Cat sprite

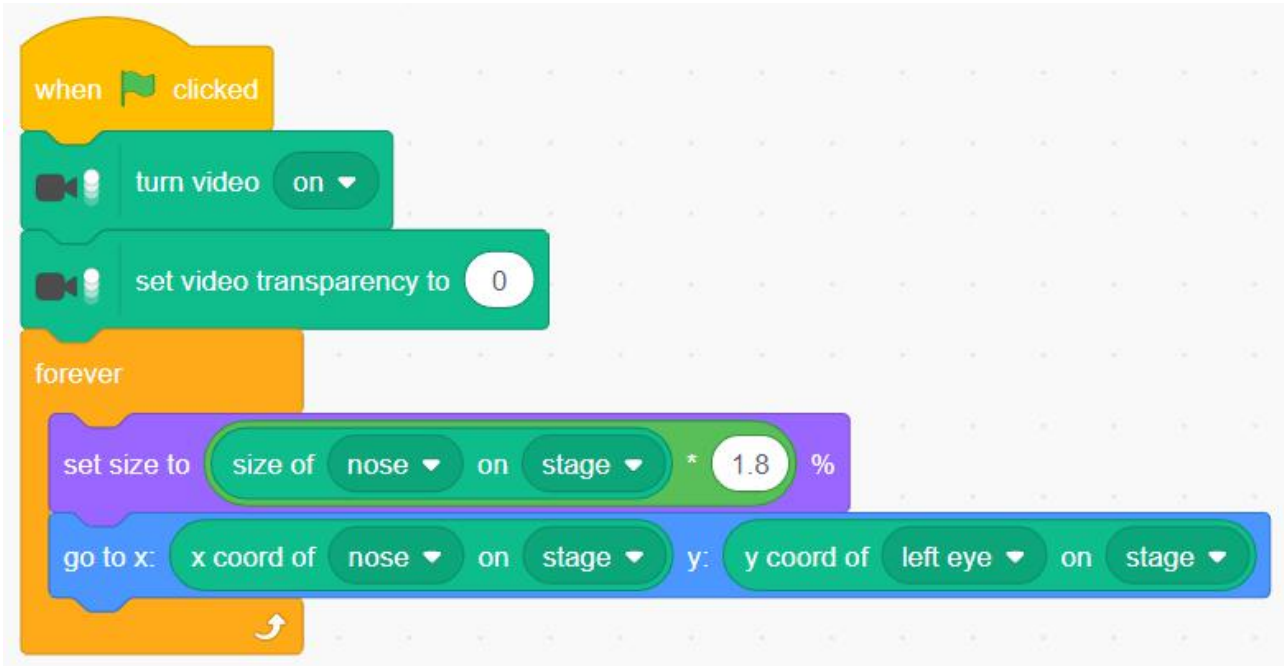
Step 5: Download picture from link https://toppng.com/transparent-glasses-PNG-free-PNG-Images_110945

Step 6: Upload picture to Scratch as sprite, rename it to "glasses"

Step 7: Sprite code

Step 8: Start the program and move your head

Step 9: Discuss the accuracy of the algorithm and how to improve it.



Third project with PICTOBLOX (Desktop application):

Step 1a: Since there is no online GUI available you have to install PictoBlox from:

<https://thetempedia.com/product/pictoblox/download-pictoblox/> (427 Mb)

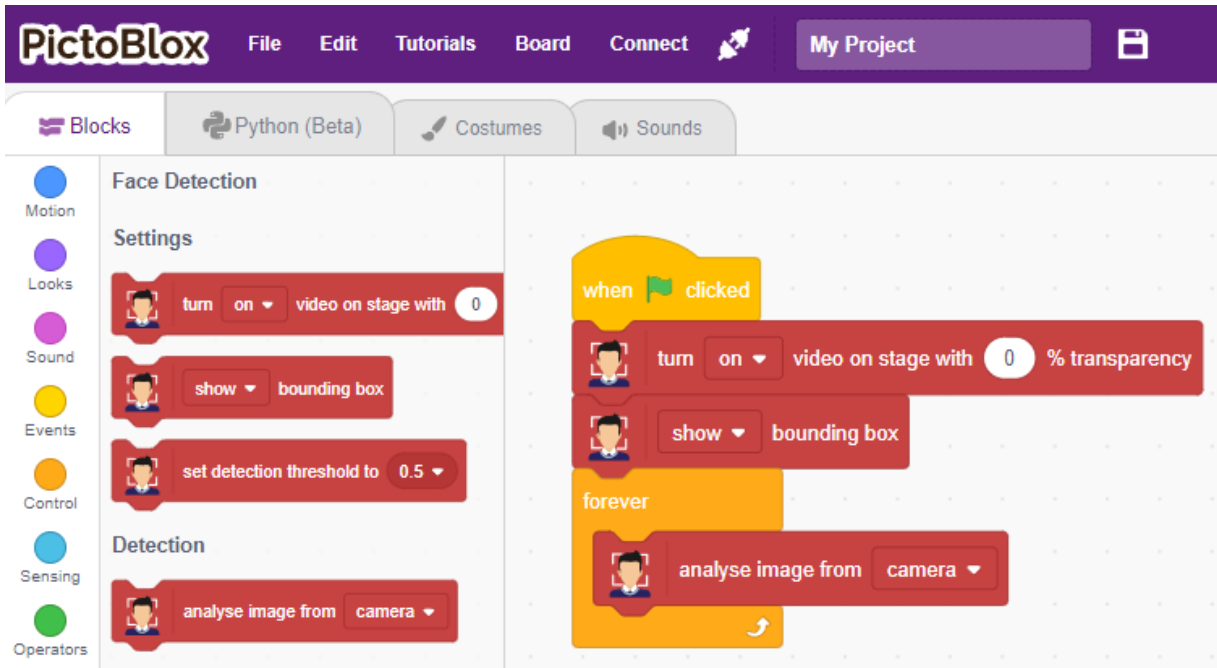
Step 2a: Open PictoBlox and choose Face detection expansion



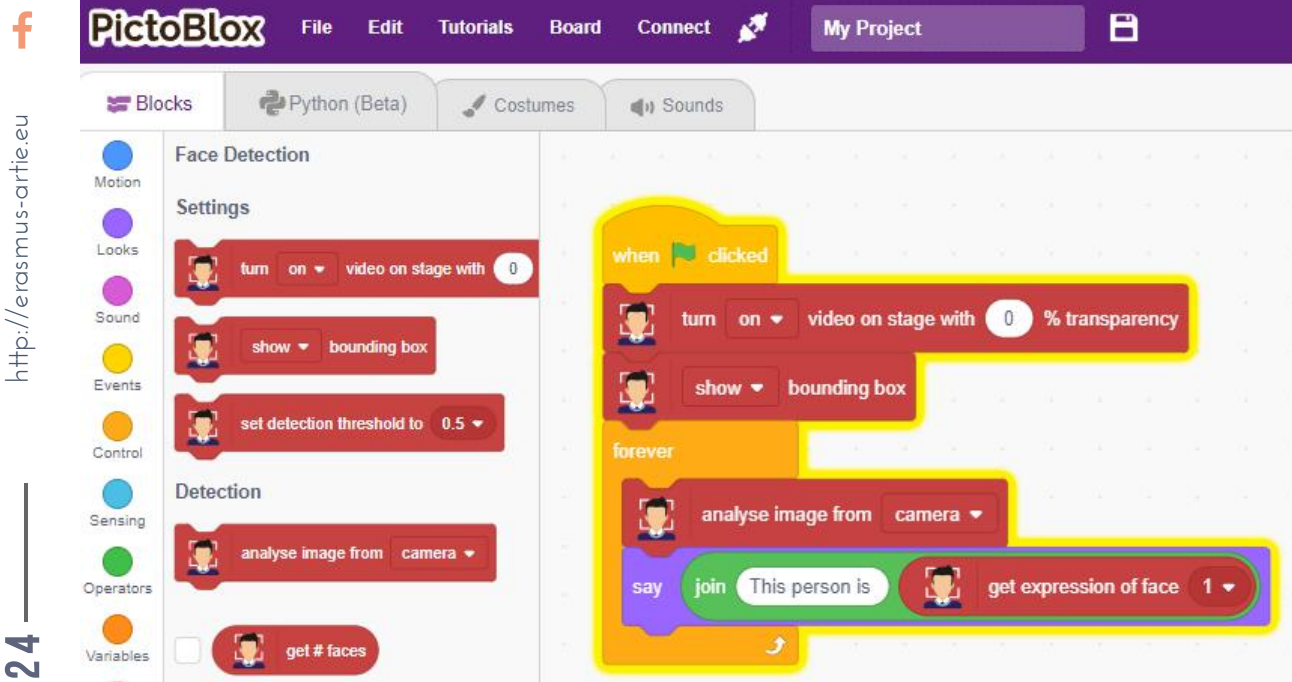
http://erasmus-artie.eu



Step 3a: Use blocks as in the picture below and it's very simple to figure out how it works. Now we have a bounding box as a block and there's no need to draw a rectangle. But the main feature is that it can detect multiple faces. Call someone to join you in front of the camera to see how it works. Check the reporter type block get # faces to see how many faces are detected.



Step 4a: And let's spice it up using join operator to display a person's face expression. You can further explore how it works with multiple faces.





Nowadays face detection software is used in almost every field from mobile devices to snap chat face filters to various security applications. Face detection helps you recognize faces, their age, expressions, gender, location, and many other features. Face detection is a broader term given to any system that can identify the presence of a human face in a visual image. Face detection has numerous applications, including in people-counting, online marketing, and even the auto-focus of a camera lens. Its core purpose is to flag the presence of a face. Face recognition has become more significant and relevant in recent years due to its potential applications. Since faces are highly dynamic and pose more issues and challenges to solve, researchers in the domain of pattern recognition, computer vision and artificial intelligence have proposed many solutions to reduce such difficulties so as to improve the robustness and recognition accuracy.

Today, face detection is used in:

Real-world applications (Amazon Rekognition: features include user verification, people counting and content moderation, often used by media houses, market analytics firms, ecommerce sites and credit solutions, BioID: GDPR-compliant solution used to prevent online fraud and identity theft, Cognitec: recognizes faces in live video streams, with clients ranging from law enforcement to border control, FaceFirst: a security solution which aims to use DigitalID to replace cards and passwords, Trueface.ai: services span to weapon detection, and are utilized by numerous sectors including education and security...)

Medical diagnoses

Criminal capture

Surveillance and compliance

CONCLUSION

Face detection is used in various sophisticated systems, and it is possible to create simple examples in tools like Scratch and PictoBlox.





Learning scenario 4

Duration : 90 min

Topics

Programming face recognition in Scratch
Interpreting outputs of face recognition algorithm

Aims

Practical use of face recognition

Outcomes

Knowing how to write a program for face recognition using Scratch
Exploring the possibilities of face recognition extension in Scratch

Programming face recognition in Scratch

The purpose of this lesson is to learn how to program facial recognition in Scratch?

Topics for discussion:

What is face recognition?
How does face recognition work?
What practical applications can it have?

Talk with your students about what they have learned about face recognition.

Do they know anything about making a program for face recognition before you lead them in?

The teacher introduces students to face recognition programming in Scratch and instructs them on how to upload images and then use these images for recognition programs.

Announcement of the goal of the lesson:

Through examples of one program, you will gain a better understanding of face recognition programs and their use.



MAIN PART

Step 1: Open your web browser and download all 20 images from: <https://bit.ly/daenerys-data>

It will be used to train Class 1

Step 2: Open your web browser and download all 20 images from: <https://bit.ly/arya-data>

It will be used to train Class 2

Step 3: Open your web browser and go to: <https://teachablemachine.withgoogle.com/>

Step 4: Click on Get started.

Step 5: Choose the Image project

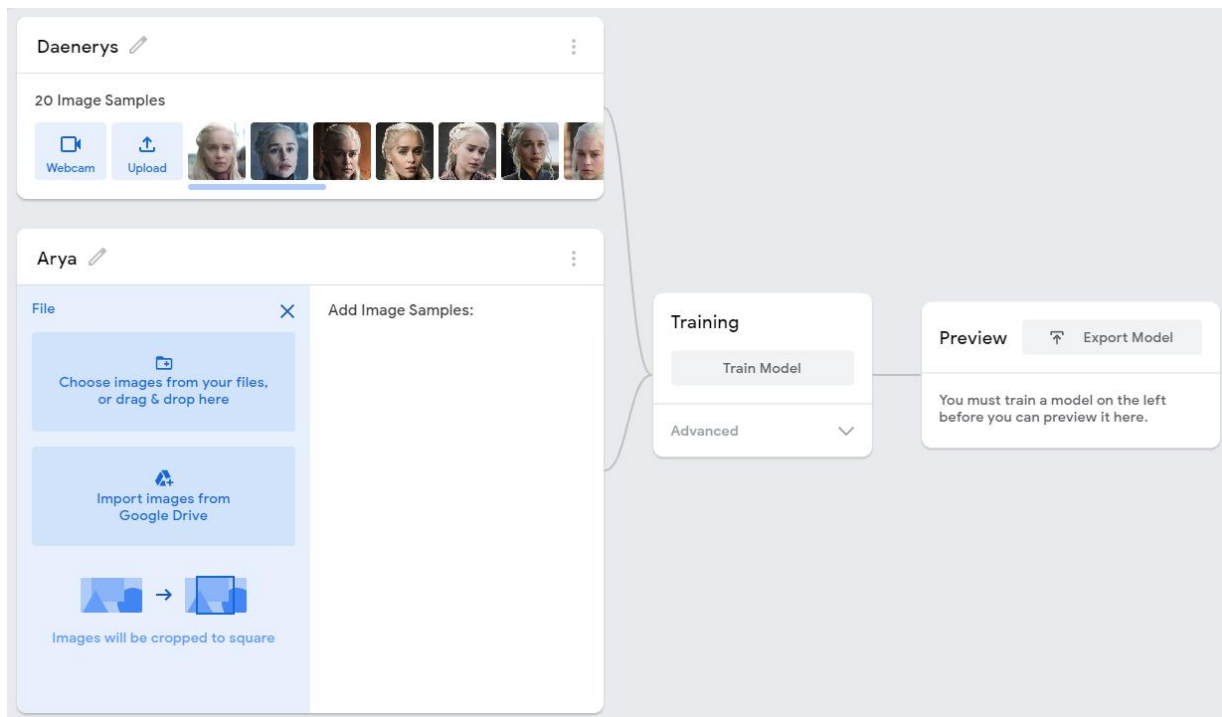
Step 6: Choose the Standard image model

Step 7: Change Class 1 name to Daenerys and Class 2 to Arya. Upload Daenerys images to Daenerys files and Arya images to Arya files as shown in the picture below



<http://erasmus-artie.eu>

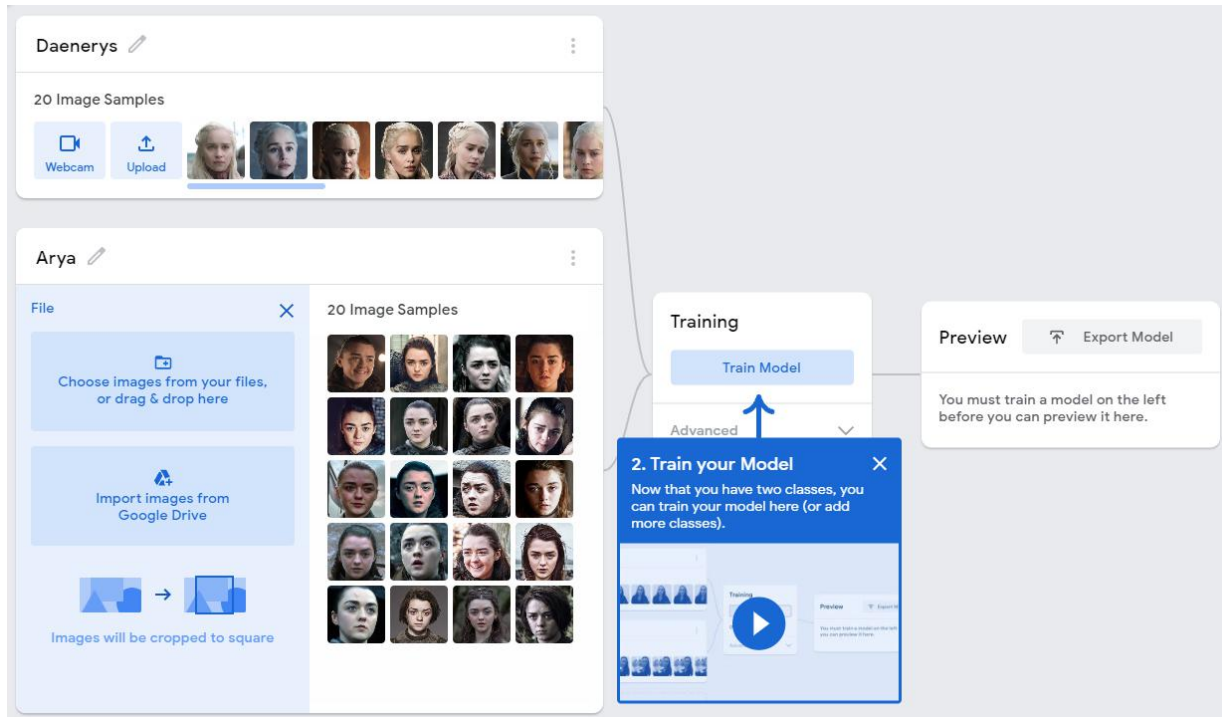
27



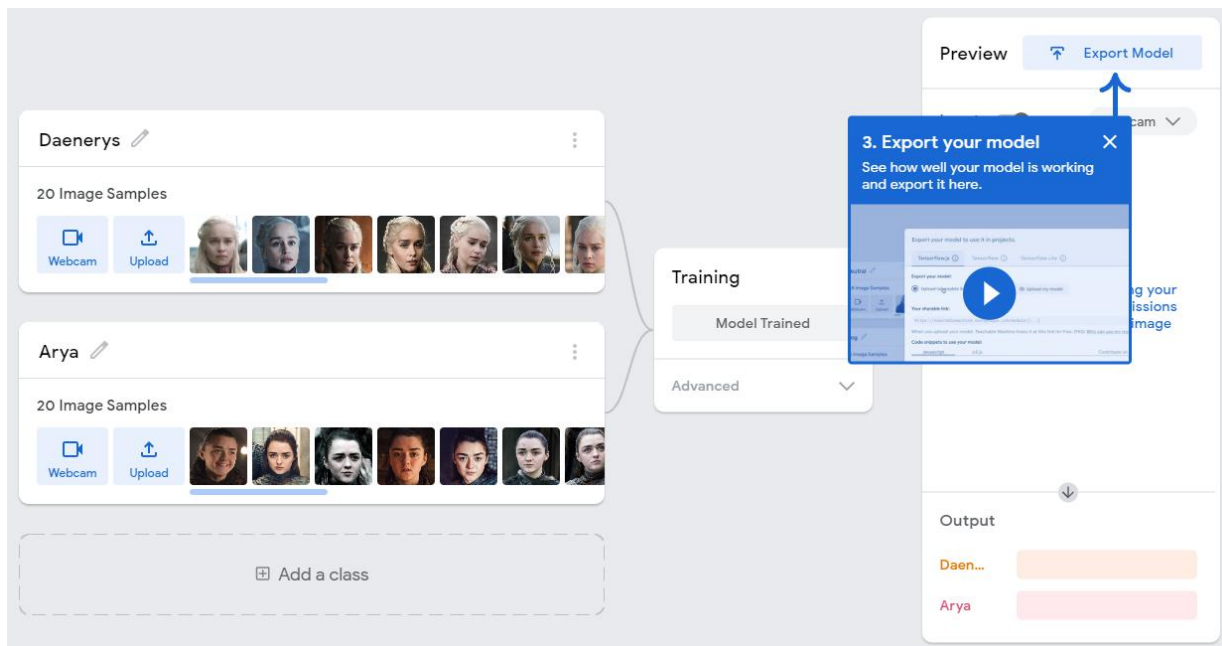
<http://erasmus-artie.eu>



Step 8: Train your model. Don't switch browser tabs during training process.

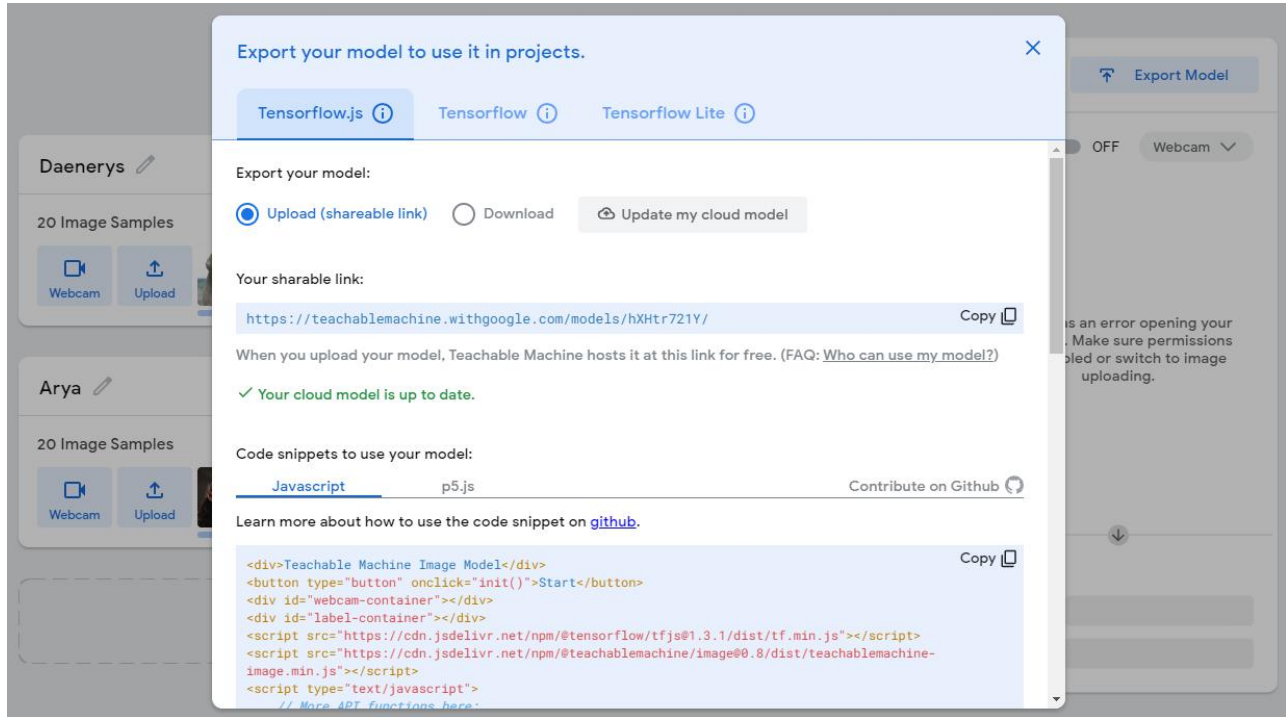



Step 9: Export your model. On the pop-up window choose to upload it to cloud (third option) and Google will host your data for free.





Step 10: Copy the link given in the text field below - this is the URL of your model. In this case it was <https://teachablemachine.withgoogle.com/models/hXHtr721Y/>



http://erasmus-artie.eu

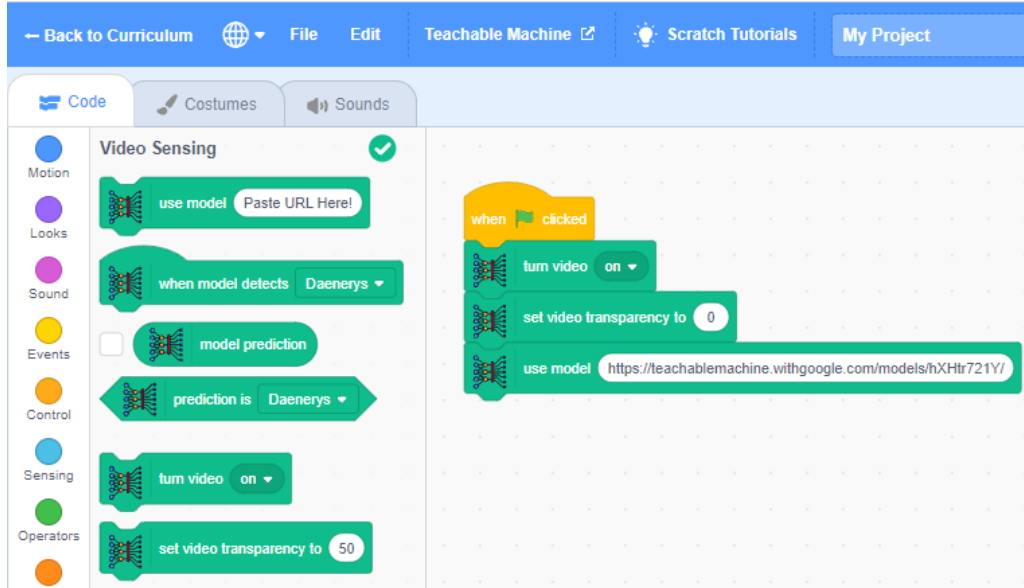
Step 11: Your model is ready to use

Step 12: Open Scratch GUI at: <https://mitmedialab.github.io/prg-extension-boilerplate/create/> and load Teachable Machine extension.



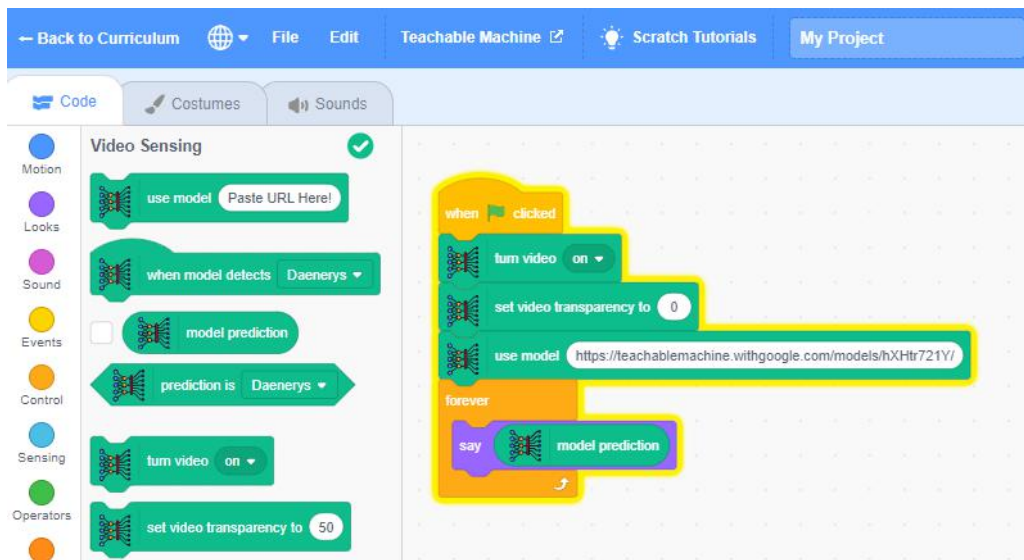


Step 13: First, we must turn on video from camera and set the transparency to 0 (non-transparent). Next, we use the model URL block and paste our model link there.



Step 14: Last few blocks are loop (forever) and say block which contains reporter type block with the prediction result. I'm pretty sure that you won't be able to get the real Daenerys or Arya in front of your camera so use your smartphone with their images and point it to the camera to see the results. You can train the model with your own images or images of your friends.

*** Make sure that you don't take anyone's photo without their permission.





PICTOBLOX (Desktop application):

Step 1a: Since there is no online GUI available, you have to install PictoBlox from:
<https://thestempedia.com/product/pictoblox/download-pictoblox/> (427 Mb)

Step 2a: Open PictoBlox and choose Face detection expansion



Step 3a: Open your web browser and download all 20 images from: <https://bit.ly/daenerys-data>

It will be used to train Class 1

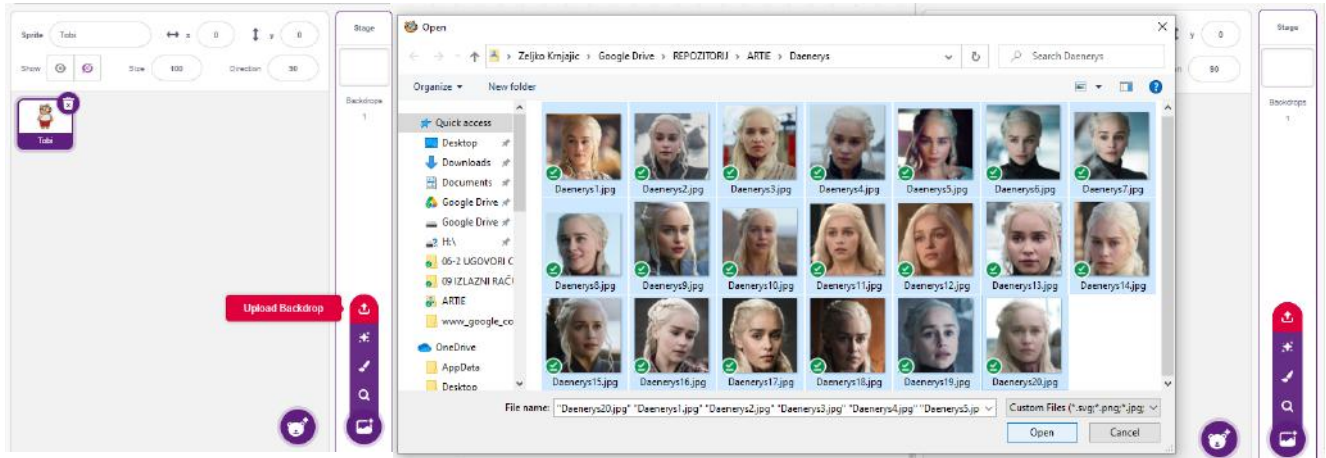
Step 4a: Open your web browser and download all 20 images from: <https://bit.ly/arya-data>
It will be used to train Class 2

Step 5a: Hide Tobi sprite from stage by clicking on icon as shown on picture below.

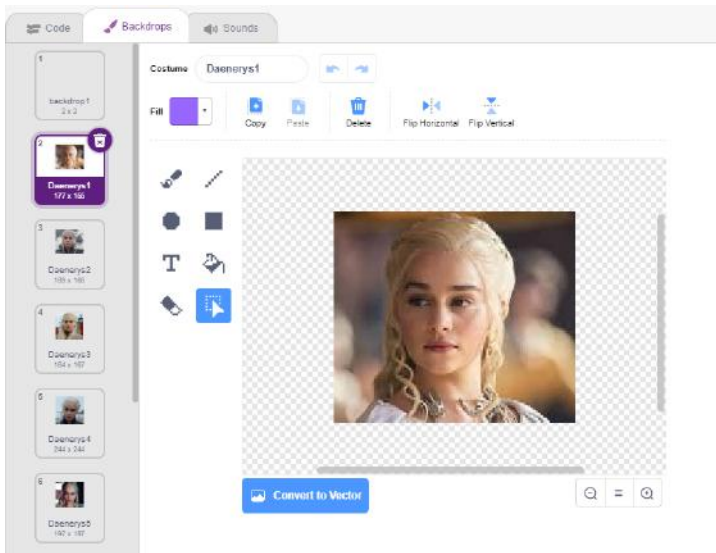




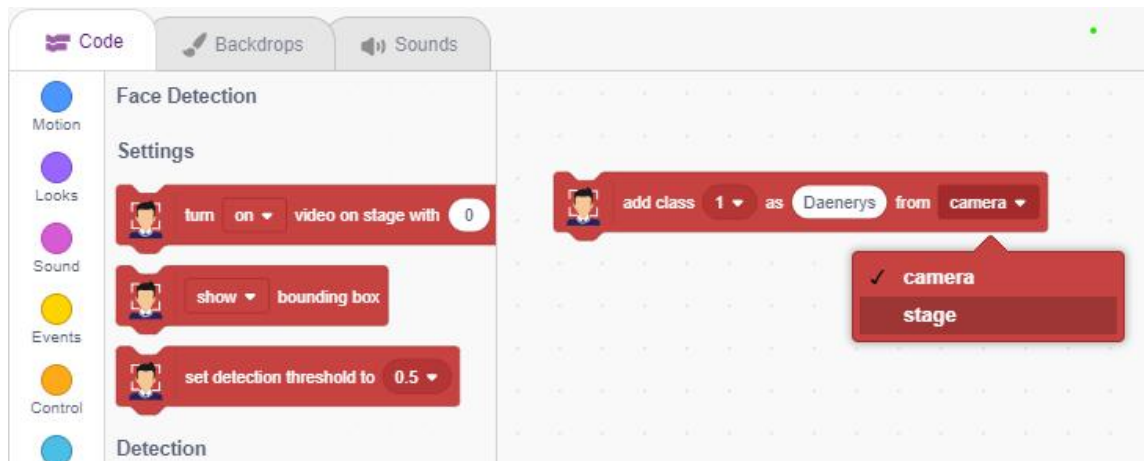
Step 6a: Upload all Daenerys images to backdrop (Upload backdrop - Select all images - Open)



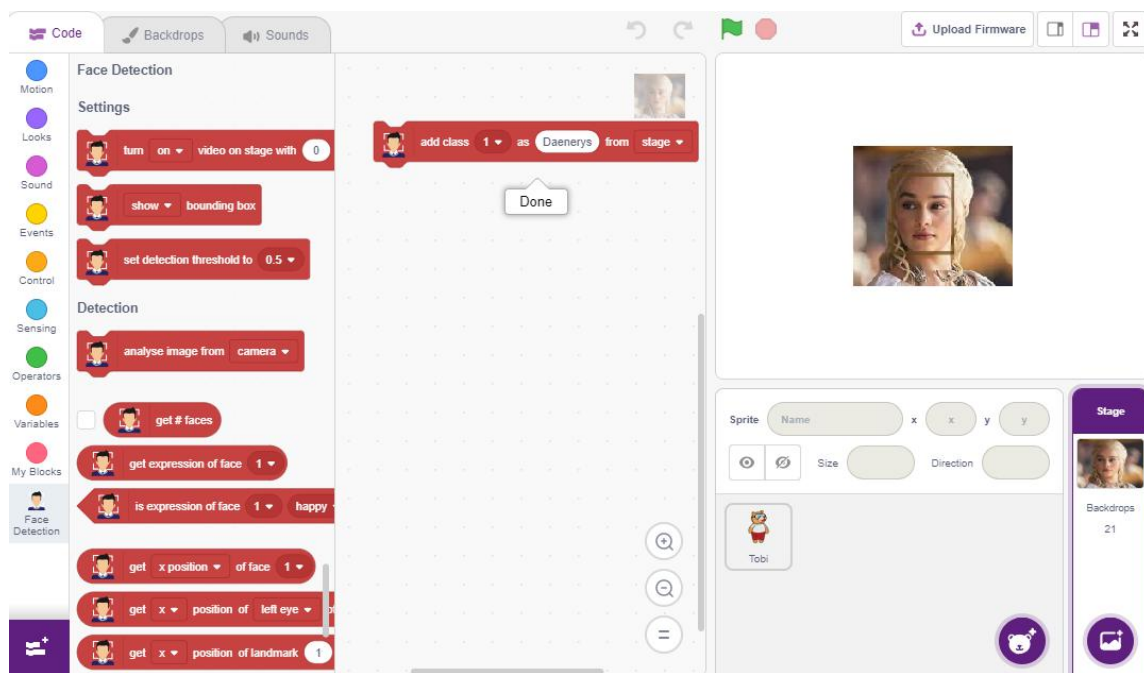
Step 7a: Select Daenerys1 image and switch to Code tab



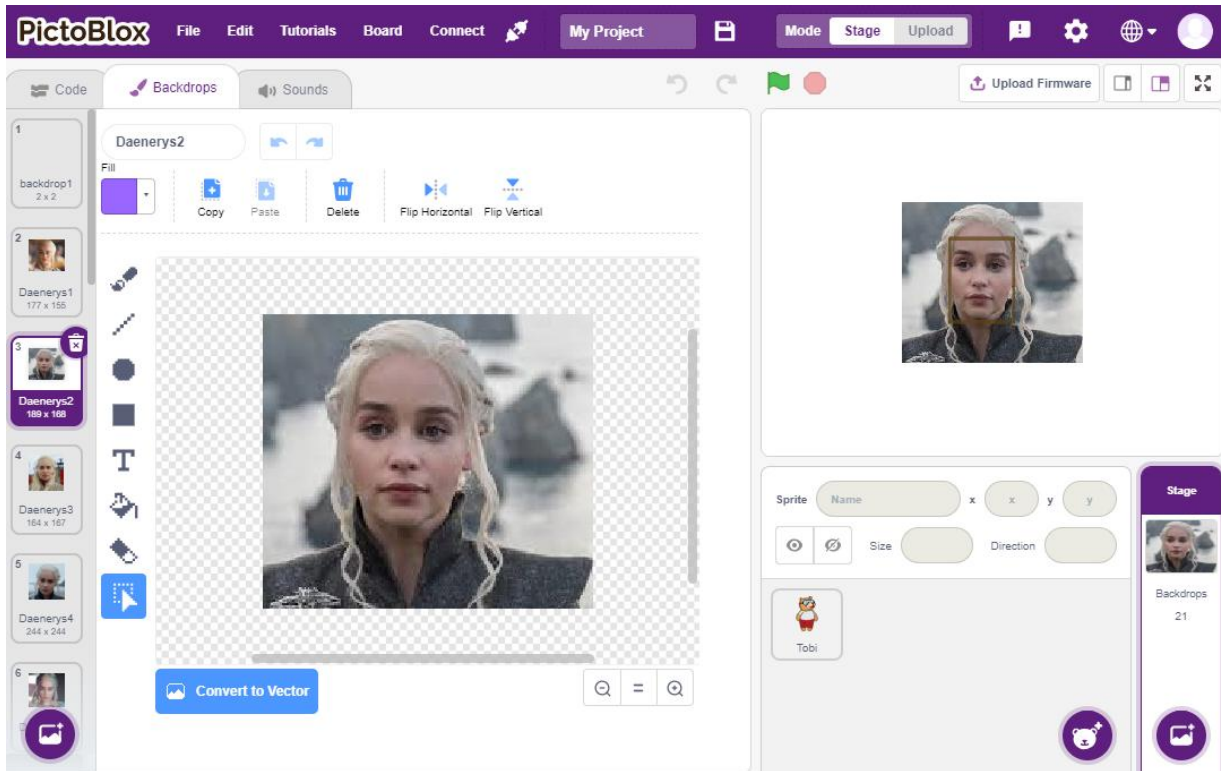
Step 8a: In Face Detection group search for add class block, drag & drop it in code area and rename class name from Jarvis to Daenerys and change source from camera to stage as shown in picture below.



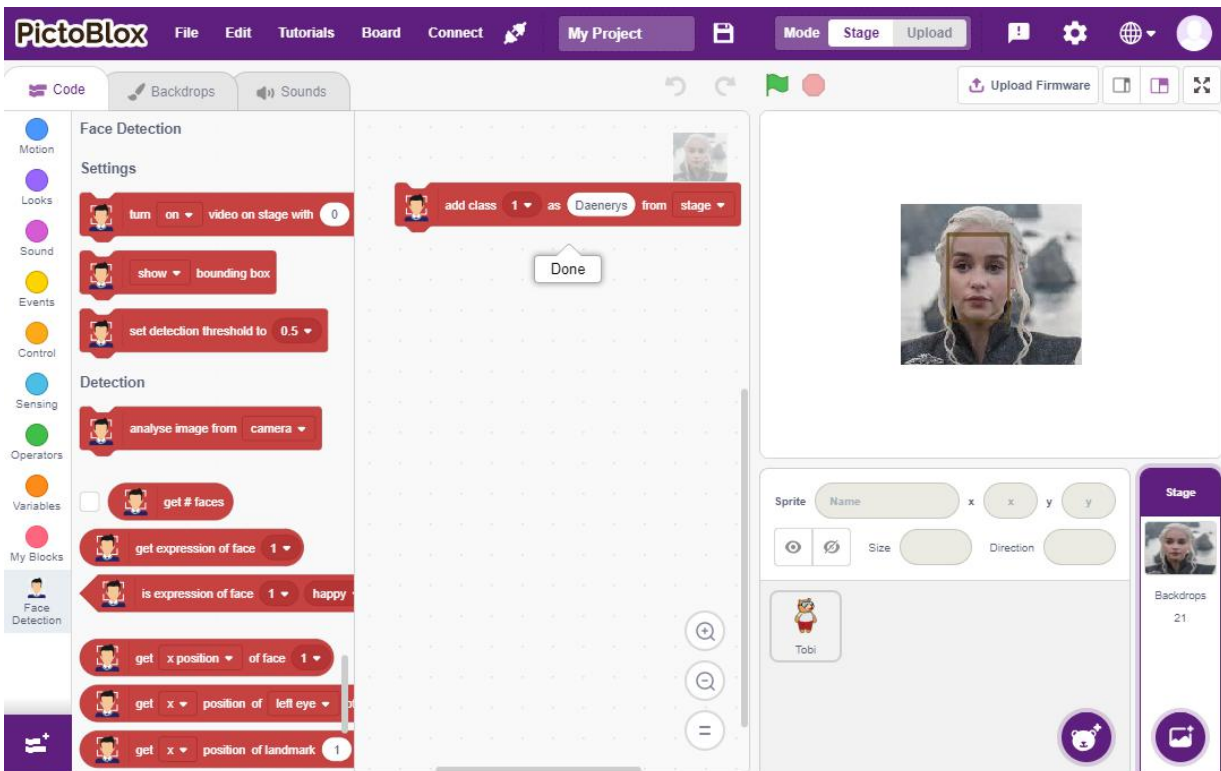
Step 9a: After changes have been made - just click on add class block to start training. You will receive a “Done” message and see the bounding box on Daenerys’s face. Training with the 1st image is completed, and you will have to repeat these steps for every other image.



Step 10a: Switch back to Backdrops tab and select Daenerys 2 image.



Step 11a: Switch back to Code tab and click on add class block again

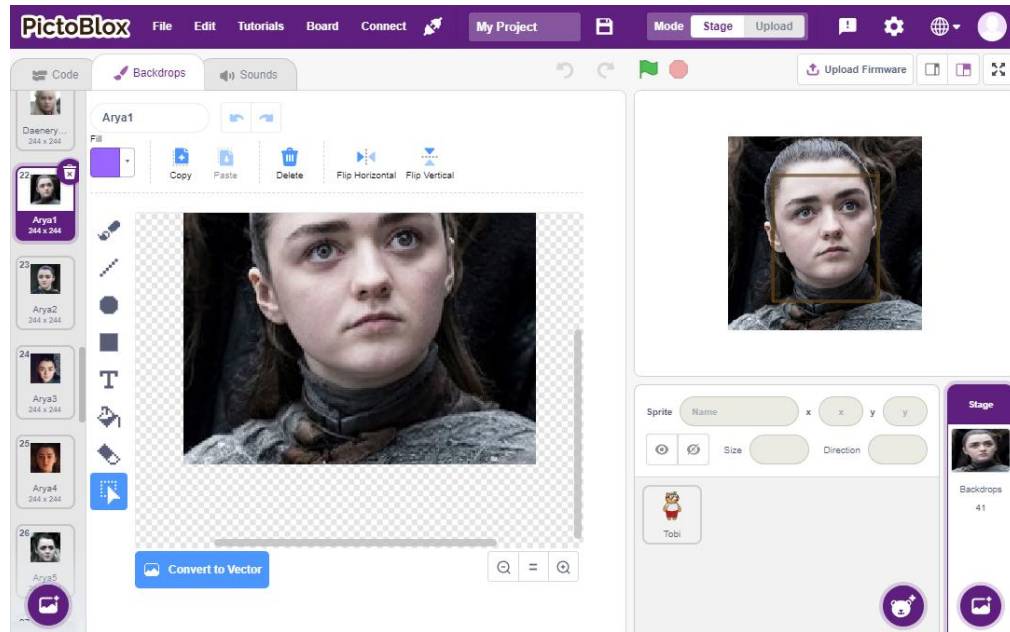




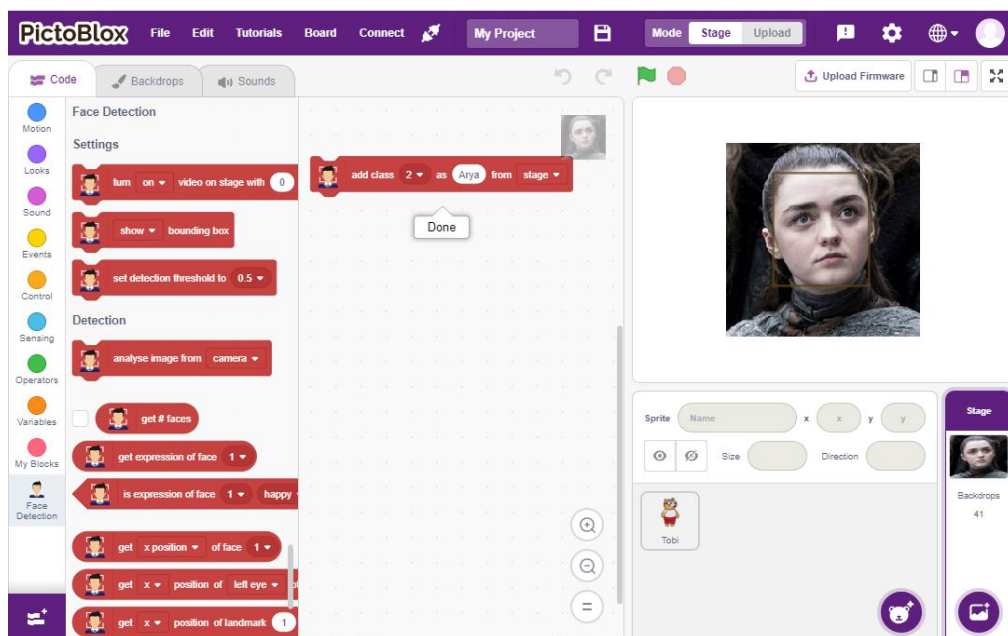
Step 12a: Repeat Steps 10a and 11a for every single image (up to Daenerys20)

Step 13a: Now upload all Arya images to backdrop same way you did in Step 6a

Step 14a: Select Arya1 image



Step 15a: Switch back to the Code tab and change (this is important since we are training 2nd class) the add class name from Daenerys to Arya. Source remains the same - from the stage. After that click on that block to train the 1st image of Arya class.





Step 16a: Switch to Backdrops tab and select Arya2 image

Step 17a: Switch to Code tab and click on add class block

Step 18a: Repeat Steps 16a and 17a for every single image (up to Arya20)

Step 19a: Your model is now ready for testing. Select Tobi sprite and make it visible (show). Set Tobi size to 30% and move it from centre to corner.

Step 20a: Plug in your camera (if you don't have one) and start coding. Turn on video with 0% transparency and show the bounding box. Next block is forever loop and Tobi will show the face recognition result based on face recognition result. It is a double if-else block and the last else case will return an empty string if no face is recognized.

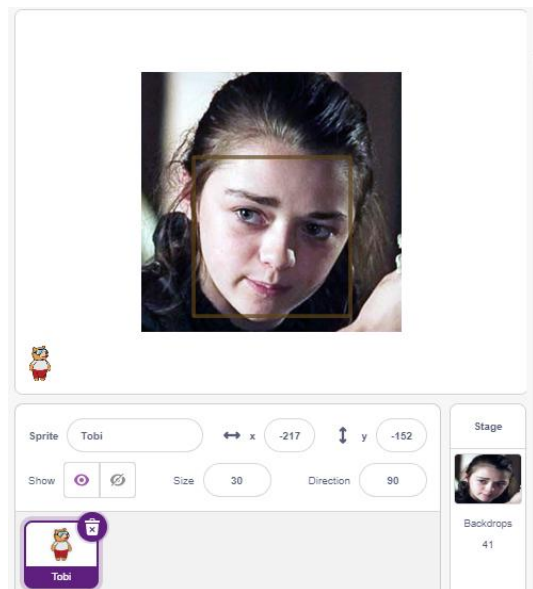
Step 21a: Start the program. Use your smartphone with images of Daenerys or Arya and point it to the camera to see the results. You can train the model with your own images or images of your friends.

*** Make sure that you don't take anyone's photo without their permission.



```

when green flag clicked
  turn on video on stage with 0 % transparency
  show bounding box
  forever
    analyse image from camera
    do face matching on camera
    if is 1 class detected then
      say Daenerys
    else
      if is 2 class detected then
        say Arya
      else
        say 
  
```





PROJECT WITH FACE RECOGNITION

Here is a little project which is basically a use case of face recognition. We have prepared a trained model of 8 female celebrities for you in Teachable Machine to compare it with your face. We have Adriana Lima, Emilia Clarke, Gal Gadot, Natalie Portman, Selena Gomez, Emma Stone, Zoe Saldana, Maisie Williams. You'll have to find out which face from the list has most similarities with you.

Step 1b: Open Scratch GUI at:

<https://mitmedialab.github.io/prg-extension-boilerplate/create/> and load Teachable Machine extension



Step 2b: First, we set the sprite size and position and then turn on video from the camera and set the transparency to 0 (non-transparent). Next, we use the model URL block and paste this model link: <https://teachablemachine.withgoogle.com/models/smuBDQTuY/>





Step 3b: Next, we'll use an event type block which is face matching triggered. And just add a say block "You look like..." to it.

```
when clicked
  set size to 50 %
  go to x: -200 y: -120
  turn video on
  set video transparency to 0
  use model https://teachablemachine.withgoogle.com/models/smuBDQTuY/

  when model detects Adriana
    say You look like Adriana Lima
```

Step 4b: Add this combination of blocks for each celebrity in the list.

```
when clicked
  set size to 50 %
  go to x: -200 y: -120
  turn video on
  set video transparency to 0
  use model https://teachablemachine.withgoogle.com/models/smuBDQTuY/

  when model detects Adriana
    say You look like Adriana Lima

  when model detects Emilia
    say You look like Emilia Clarke

  when model detects Gal
    say You look like Gal Gadot

  when model detects Natalie
    say You look like Natalie Portman

  when model detects Selena
    say You look like Selena Gomez

  when model detects Emma
    say You look like Emma Stone

  when model detects Zoe
    say You look like Zoe Saldana

  when model detects Maisie
    say You look like Maisie Williams
```





Step 5b: Plug in/enable your web camera, start the code, and see who your match is!

Step 6b: Want to change the current model with your own? There is a great celebrity images dataset at: <https://www.kaggle.com/hereisburak/pins-face-recognition>

You may be asked to register with Kaggle before you can download it. Or you may collect images manually via Google search. After you collect all the images you want - use the Teachable machine at: <https://teachablemachine.withgoogle.com/> to train your model the way you did in **Programing face recognition in Scratch** scenario.

Face recognition is a technology capable of identifying or verifying a subject through an image, video, or any audiovisual element of his face. Generally, this identification is used to access an application, system, or service. Before we start programing face recognition, we must collect the photos of specific face and train the model. We have used an application called Teachable Machine. Teachable Machine is a web-based tool that makes creating machine learning models fast, easy, and accessible to everyone.

Did you notice that this project is completely web based and no software installation is needed?

Today, many consumers worldwide regularly interact with facial recognition technology.

Key Functionalities of Facial Recognition:

Identity Verification:

Identify individuals and apply specified rules based on the category they fall into, for example, VIP, registered visitor, block listed, employee, or student. Use the information to enhance and automate processes such as (1) access control, (2) security protection, (3) customer or visitor greetings, and (4) employee time clocks.

eKYC and Spoofing Prevention:

Validate a person's identity using a live photo or video capture with a scanned (and verified) ID. This is called eKYC (electronic Know Your Customer) and is widely used in BFSI or similar cases.

Authorization:

Identify whether an individual is in a pre-authorized database to (1) withdraw cash from an ATM, (2) access a medical cabinet containing secured drugs, or (3) unlock expensive machinery requiring well-trained operators.

Customer Segmentation and Analytics:

For smart advertising, analyse the characteristics of a person standing in front of a digital sign, such as gender, age, and emotion.





Health Measures:

Confirm that a person is properly wearing a mask, as well as verifying they do not have a fever before granting access to a building or restaurant.

When looking at vertical markets, 10 industries stand out as being ripe for integrating facial recognition and, in many cases, are already embracing it:

1. Manufacturing and warehousing
2. Banking, Financial Services and Insurance (BFSI)
3. Smart offices
4. Smart homes and residential complexes
5. Retail
6. Public transportation and airports
7. Healthcare facilities
8. Schools and universities
9. Hospitality
10. Restaurants and bars

CONCLUSION

Facial recognition is used in various sophisticated systems, and it is possible to create simple examples in tools like Scratch and PictoBlox.





Learning scenario 5

Duration: 90 min

Topics

Object detection and classification for beginners in Scratch

Aims

To learn what is object detection and classification for beginners in Scratch

Outcomes

Understanding what object detection and classification is with the help of simple example in Scratch

Understanding the difference between Object detection and Object classification



Object detection and classification for beginners in Scratch

Object detection and classification have attracted much attention in the past decades. In the computer vision field, one of the most common questions has to do with the difference between image classification, object detection and image segmentation.

Announcement of the goal of the lesson:
An introduction to object detection and classification for beginners through practical application examples.

Let's start with understanding of what image classification is:
Consider the image below:





MAIN PART

You recognize it instantly. It's a dog. Take a step back and analyse how you came to this conclusion. You were shown an image and you classified the class it belonged to (a dog, in this instance). And that, in a nutshell, is what Image Classification is all about.

As you saw it, there's only one object here: a dog. We can easily use an image classification model and predict that there's a dog in the given image. But what if we have both a cat and a dog in a single image



We can train a multi-label classifier in that instance. But we won't know the location of either animal/object in the image. That's where Image Localization comes into the picture. It helps us to identify the location of a single object in the given image. In case we have multiple objects present, we then rely on the concept of [Object Detection](#).

We can predict the location along with the class for each object using OD.



<http://erasmus-artie.eu>

Classification



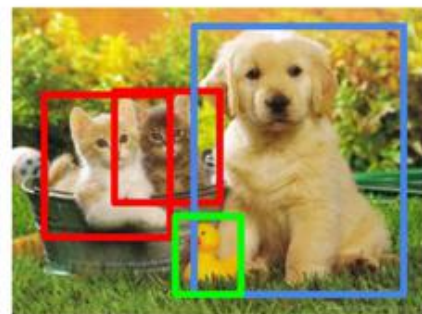
CAT

Classification + Localization



CAT

Object Detection



CAT, DOG, DUCK

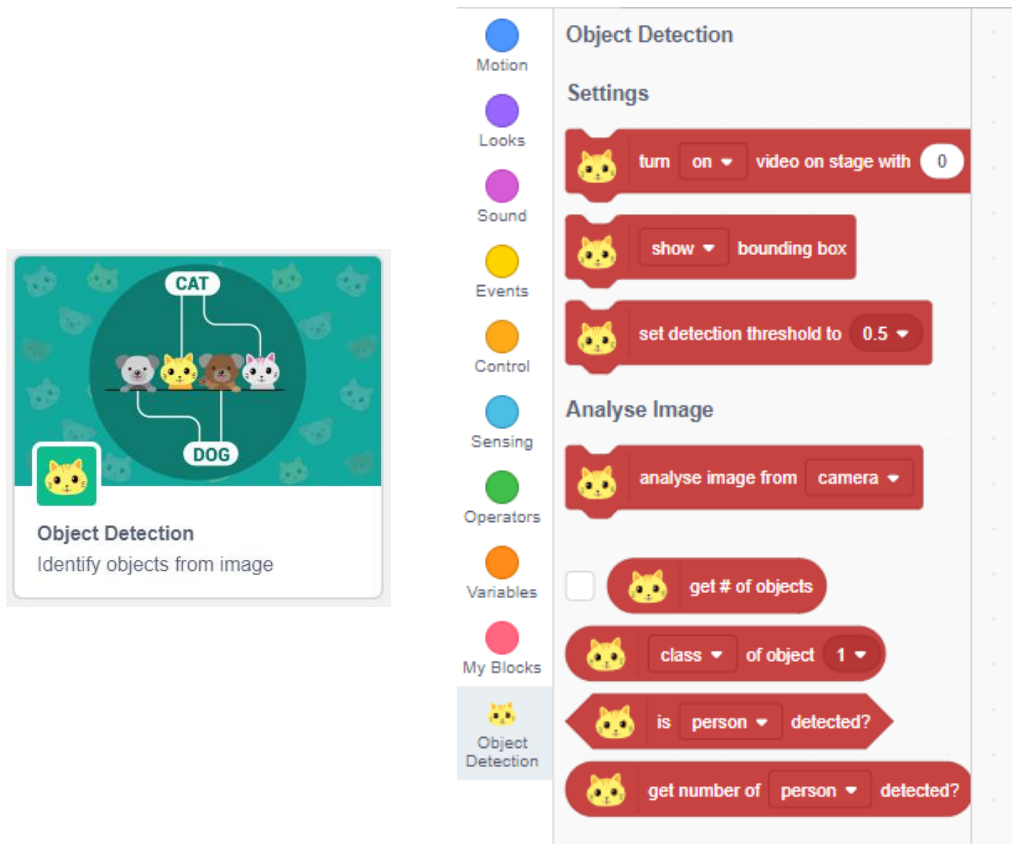


Application with object detection

PictoBlox is currently the only application with object detection capability. It's a desktop type application and you must install it first from <https://thetempedia.com/product/pictoblox/download-pictoblox/> (427 Mb)

Step 1: After installation load Object detection extension

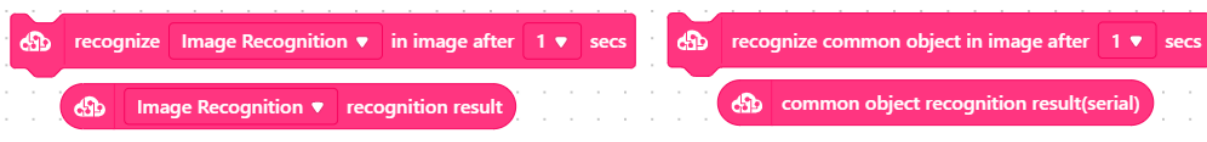
Step 2: These are the blocks available for coding.



First 3 blocks are camera stream settings, and the others are used for analysing and reporting. Situation with object classification is much better since we have few more applications to use (beside PictoBlox)

Makeblock (mBlock) - <https://ide.mblock.cc/>

Load Cognitive services extension and you will find 4 blocks to use for object recognition (classification).



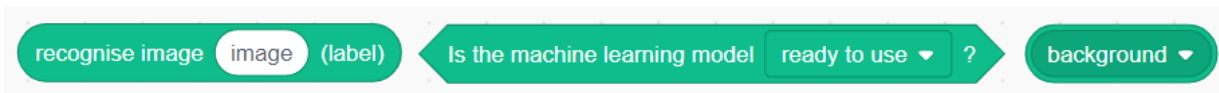


Makeblock also provides Teachable machine extension (not to be confused with Google's) where you can train up to 3 classes and use it for object classification.

Scratch (ML4KIDS) - <https://machinelearningforkids.co.uk/scratch3/>
Load Imagenet extension.



3 blocks are available. Use it in combination with Video sensing extension to turn camera video on/off and set transparency. It has been trained to recognize photos of one-thousand common objects. The machine learning model is based on MobileNet (a ML model designed for mobile devices, so it doesn't need much computing power). Complete list of objects is available here: <https://storage.googleapis.com/download.tensorflow.org/data/ImageNetLabels.txt>

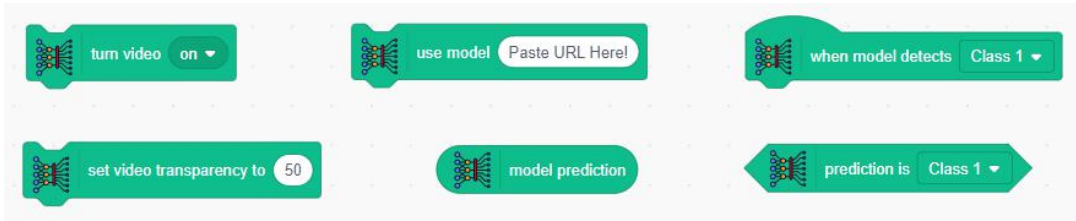


Scratch (MITMEDIALAB) - <https://mitmedialab.github.io/prg-extension-boilerplate/create/>
Load Teachable machine extension



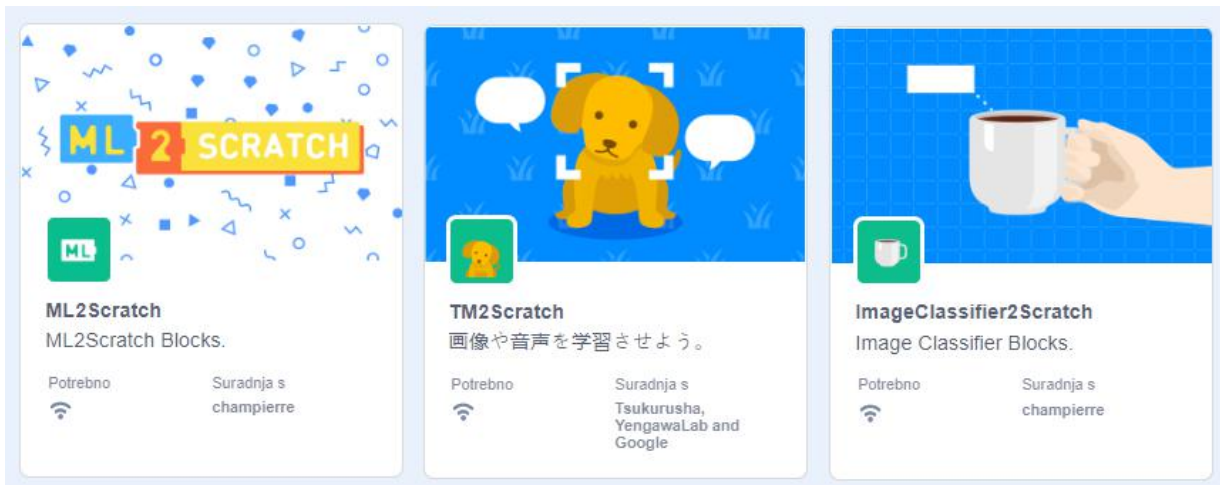


Use these blocks in combination with *Google Teachable machine*.



Stretch3 (github.io) - <https://stretch3.github.io/>

Load *ML2Scratch*, *TM2Scratch* and *ImageClassifier2Scratch* extensions to use plenty blocks for object classification and training



<http://erasmus-artie.eu>

ML2Scratch blocks





TM2Scratch blocks



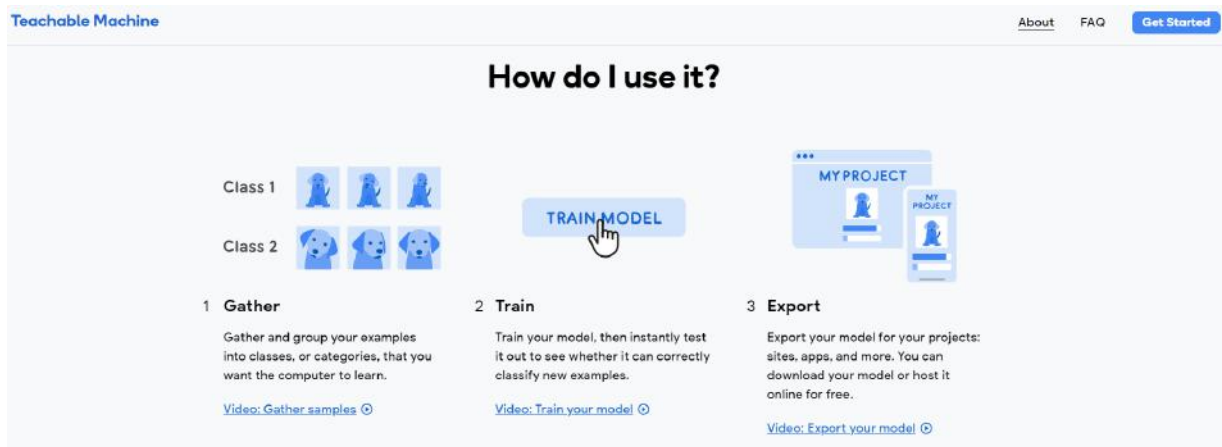
ImageClassifier2Scratch blocks



Teachable machine (Google) - <https://teachablemachine.withgoogle.com/>
This application is used to train your model and use it for object recognition in combination with Teachable machine extensions available in PictoBlox, Scratch (MITMEDIALAB) and Stretch3



<http://erasmus-artie.eu>

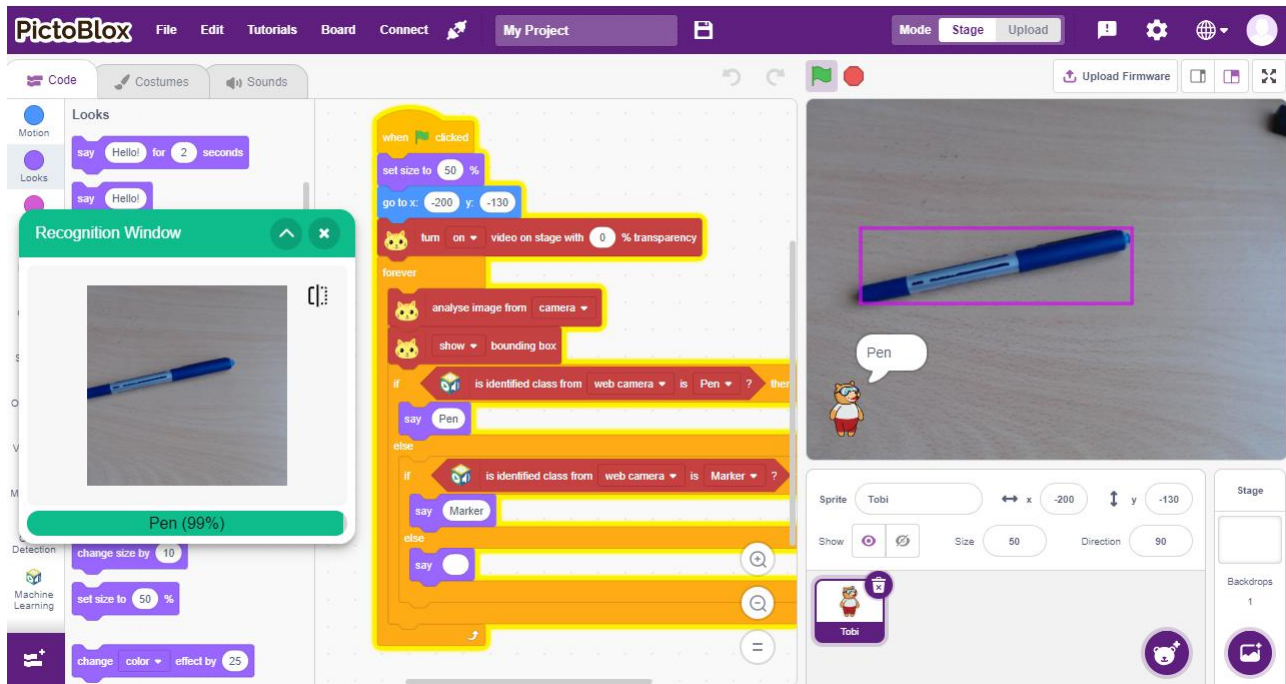


Use case for object detection - pen or marker.

Teachable machine is used for training 2 classes. Link of a trained model is:

<https://teachablemachine.withgoogle.com/models/FdWn0CA2a/>

Extension used in PictoBlox are Object detection and Machine learning.



Object detection and classification are two key tasks for image understanding. Recognizing objects in an image requires combining many different signals from the raw image data. Two kinds of information are often used: the local appearance that describes the object itself and the global representation that captures the image specific information. These two types of information are often used in two tasks: object detection and classification.

Classification

Classification is a machine learning task for determining which objects are in an image or video. It refers to training machine learning models to recognize which classes (objects) are present. Classification is useful at the yes-no level of deciding whether an image contains an object/anomaly or not.

A separate task from classification is localization or determining the position of the classified objects in the image or video.



f

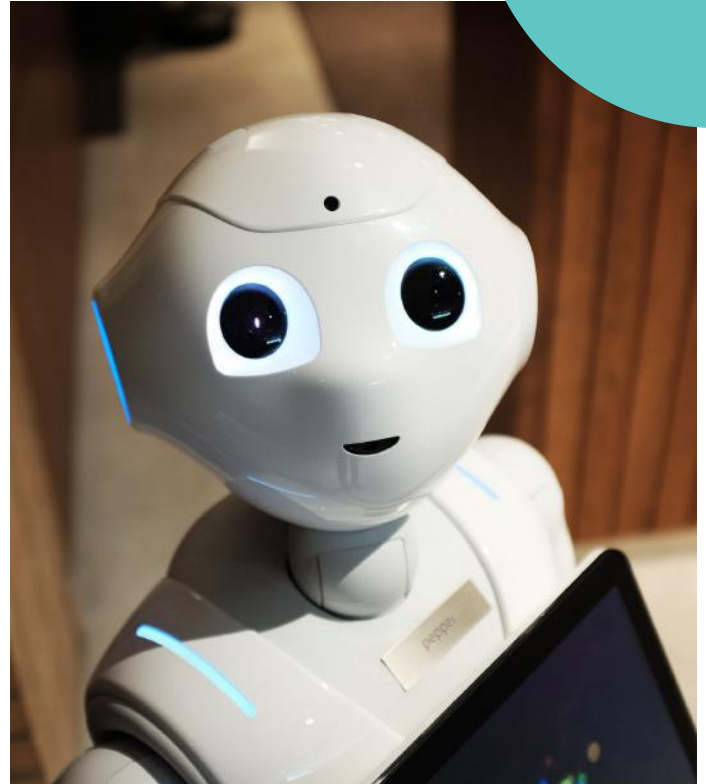
<http://erasmus-artie.eu>



Object detection

Object detection combines classification and localization to determine what objects are in the image or video and specify where they are in the image. It applies classification to distinct objects and uses bounding boxes. Object detection is useful in identifying objects in an image or video. Use cases for object detection include facial detection with any post-detection analysis, for example, expression detection, age estimation or drowsiness detection. Many real-time object detection applications exist for traffic management, such as vehicle detection systems based on traffic scenes.

As described above, the most popular approaches to computer vision are classification and object detection to identify objects present in an image and specify their position.



<http://erasmus-artie.eu>

CONCLUSION

Object detection and classification are two key tasks for image analysis.



Learning scenario 6

Duration: 90 min

Topics

Programing object detection in Scratch

Aims

To learn to program object detection with uploaded examples

Outcomes

Knowing how to write a program for object detection using Scratch



Programing object detection in Scratch

In order to understand object detection, let's review what we have learned so far.

What is object detection?

How does object detection work?

Object detection is a computer vision technique whose purpose is to identify and locate objects within an image or video. Specifically, object detection draws bounding boxes around these detected objects, which allow us to locate where said objects are in (or how they move through) a given scene. Before we start programing object detection, we must collect the photos of specific faces and train the model. We will use an application called Teachable Machine. Teachable Machine is a web-based tool that makes creating machine learning models fast, easy, and accessible to everyone

Announcement of the goal of the lesson:

Understanding object detection software and its use through examples.



<http://erasmus-artie.eu>



MAIN PART

TASK: Is it a cat or a dog?

Create a model and program that will detect whether a cat or a dog is in the camera stream

WEB BASED APPLICATION (no software installation)

Step 1: Open your web browser, select, and download images from: <https://bit.ly/cats-image-dataset>

It will be used to train Class 1

Step 2: Open your web browser, select, and download images from: <https://bit.ly/dogs-image-dataset>

It will be used to train Class 2

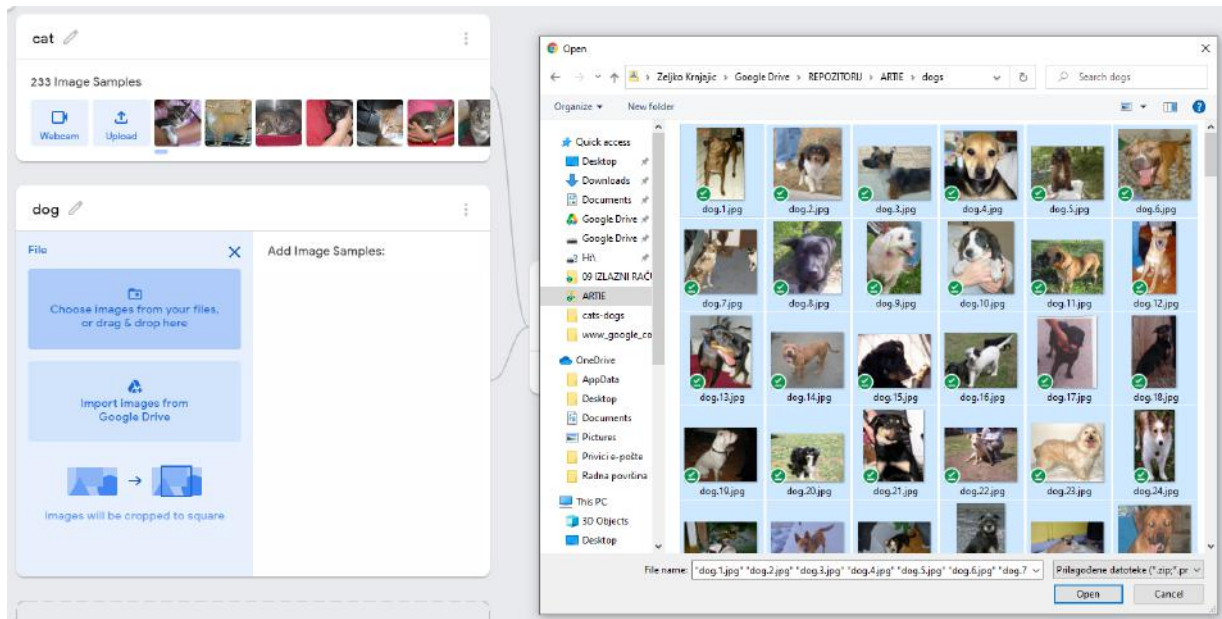
Step 3: Open your web browser and go to: <https://teachablemachine.withgoogle.com/>

Step 4: Click on Get started.

Step 5: Choose the Image project

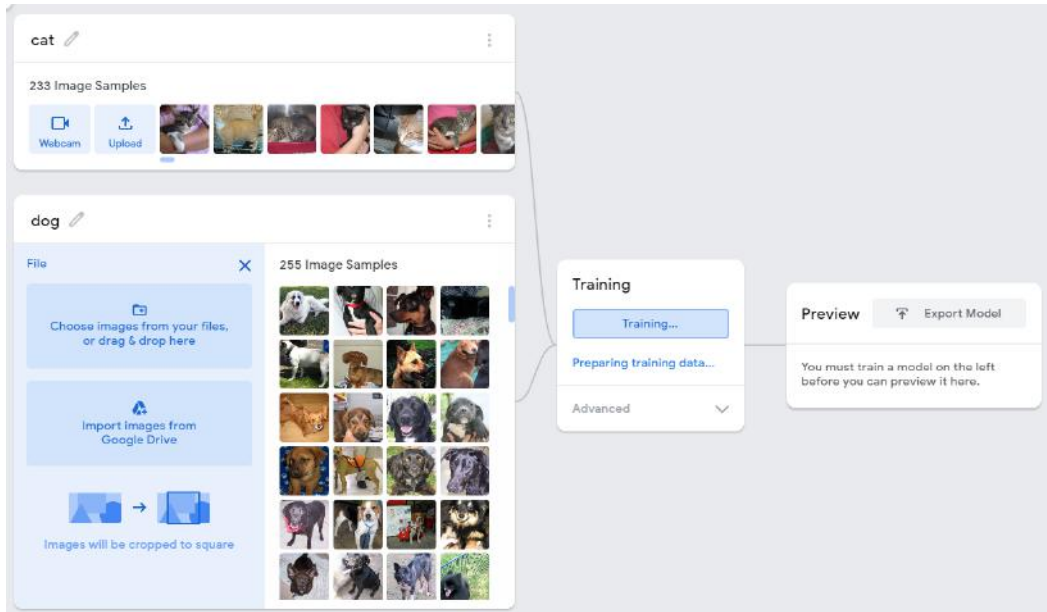
Step 6: Choose the Standard image model

Step 7: Change Class 1 name to cat and Class 2 to dog. Upload cats' images to cat files and dogs' images to dog files as shown in the picture below

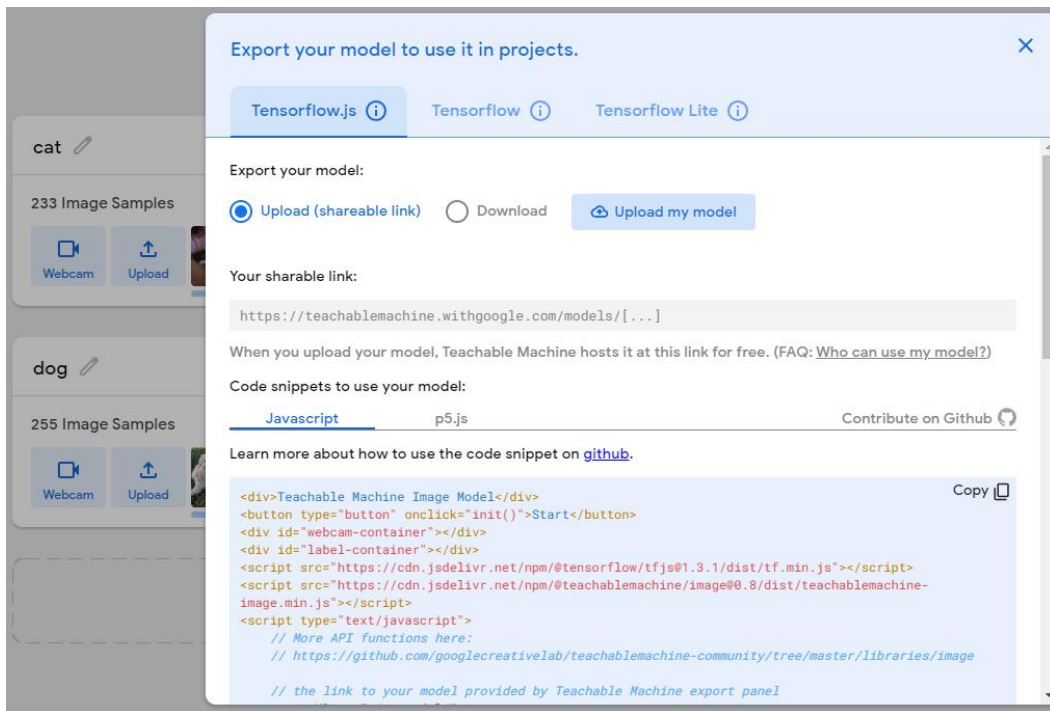




Step 8: Train your model. Don't switch browser tabs during training process.



Step 9: Export your model. On the pop-up window choose to upload it to the cloud (third option - Upload my model) and Google will host your data for free.





Step 10: Copy the link given in the text field below - this is the URL of your model.
In my case it was <https://teachablemachine.withgoogle.com/models/gS4NT1mgE/>

Upload (shareable link)
 Download

Your sharable link:

<https://teachablemachine.withgoogle.com/models/gS4NT1mgE/>

When you upload your model, Teachable Machine hosts it at this link for free. (FAQ: [Who can use my model?](#))

✓ Your cloud model is up to date.

Step 11: Your model is ready to us

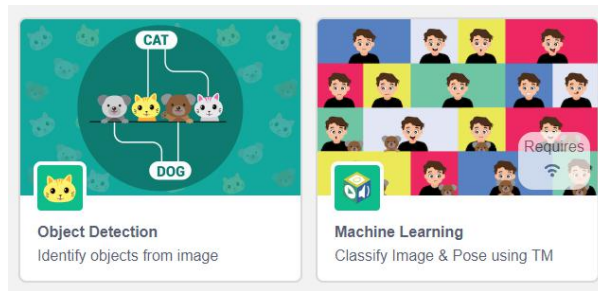
PICTOBLOX (Desktop application):

Step 1a: Download and install **PictoBlox** since it's currently the only one with object detection capability. It's desktop type application and you must install it first from <https://thetempedia.com/product/pictoblox/download-pictoblox/> (427 Mb)

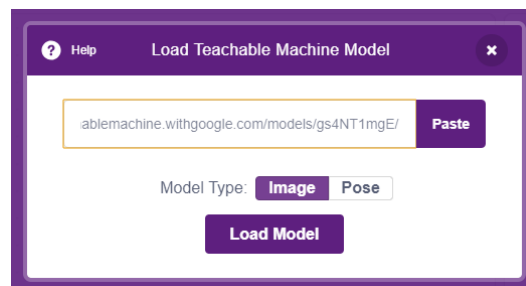
Step 2a: Load extensions Object Detection and Machine Learning.



<http://erasmus-artie.eu>

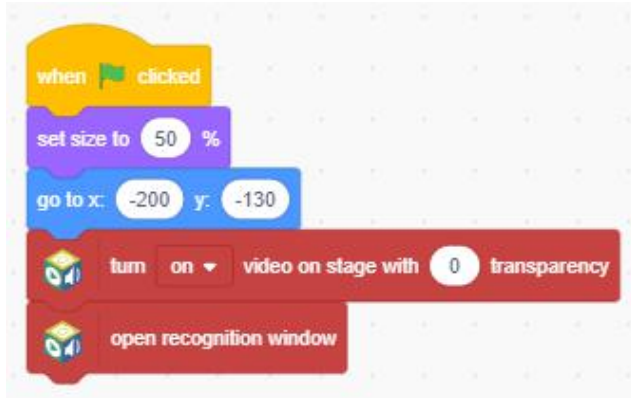


Step 3a: Select Machine Learning group and choose Load a Model. Paste the model link from teachable machine (below): <https://teachablemachine.withgoogle.com/models/gS4NT1mgE/> and click on Load Model





Step 4a: We will perform an object classification and object detection simultaneously. First, we must start the program, resize, and move the Tobi sprite in lower left corner. The next few blocks from Machine Learning extension are used to turn on the video and open recognition window. Use turn on flipped video on stage with 0 transparency if your video is flipped.

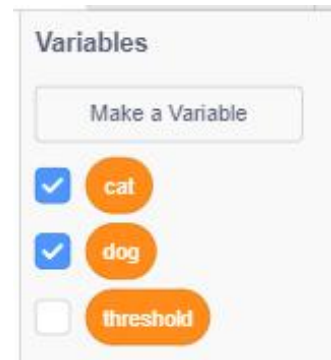
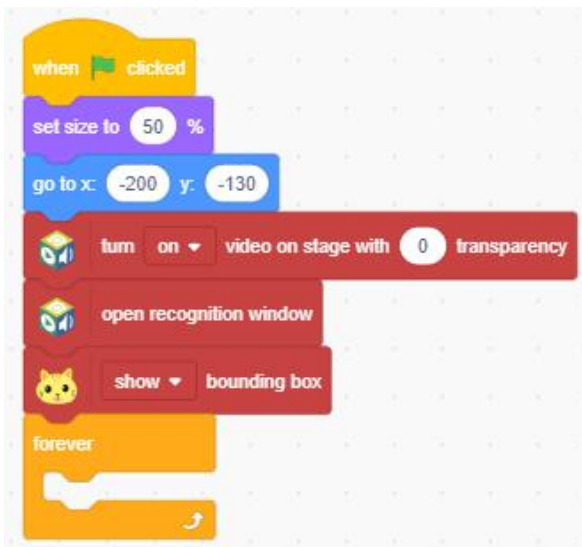


Step 5a: From Object Detection extension use show bounding box block to display the position of an object in the video stream and add a forever loop block.

Step 6a: Make 3 variables **cat**, **dog** and **threshold**. Display **cat** and **dog** on stage by checking it. Leave the threshold unchecked.



<http://erasmus-artie.eu>

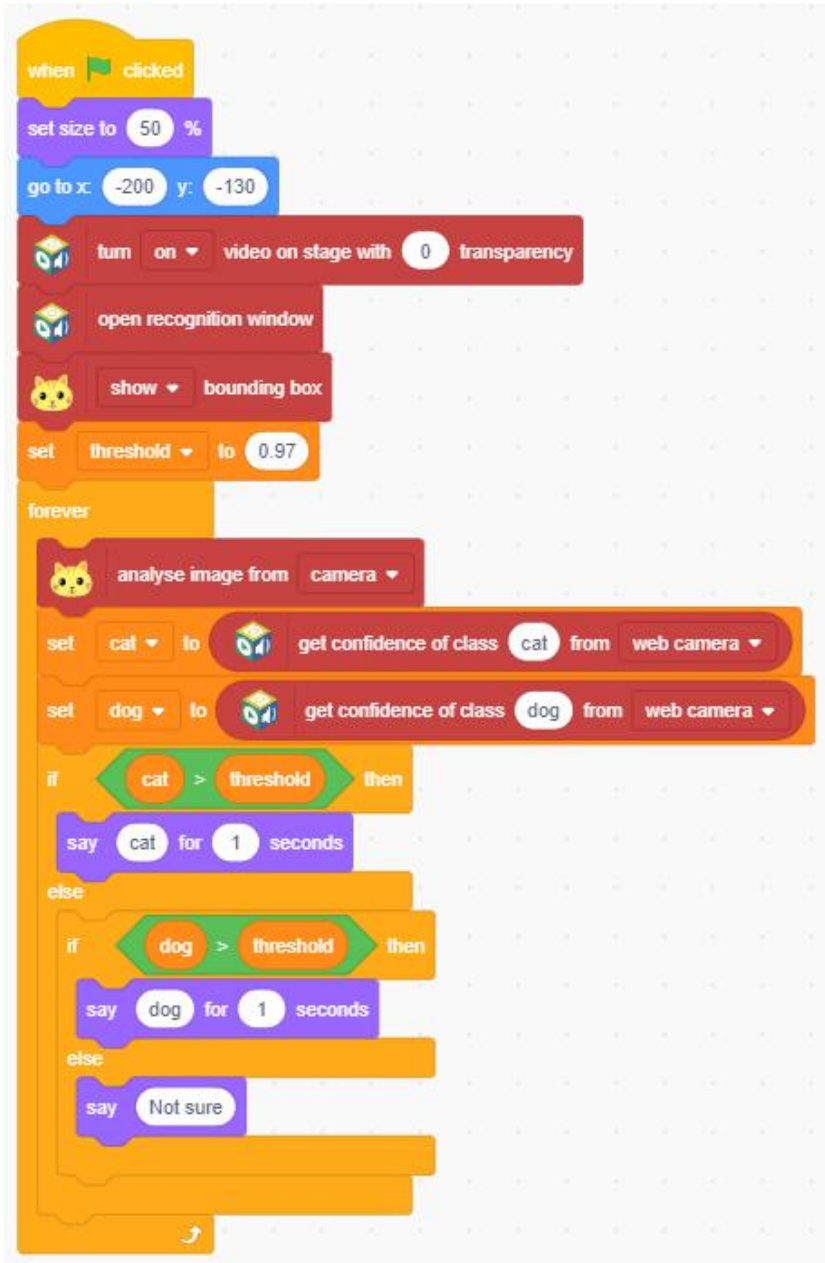




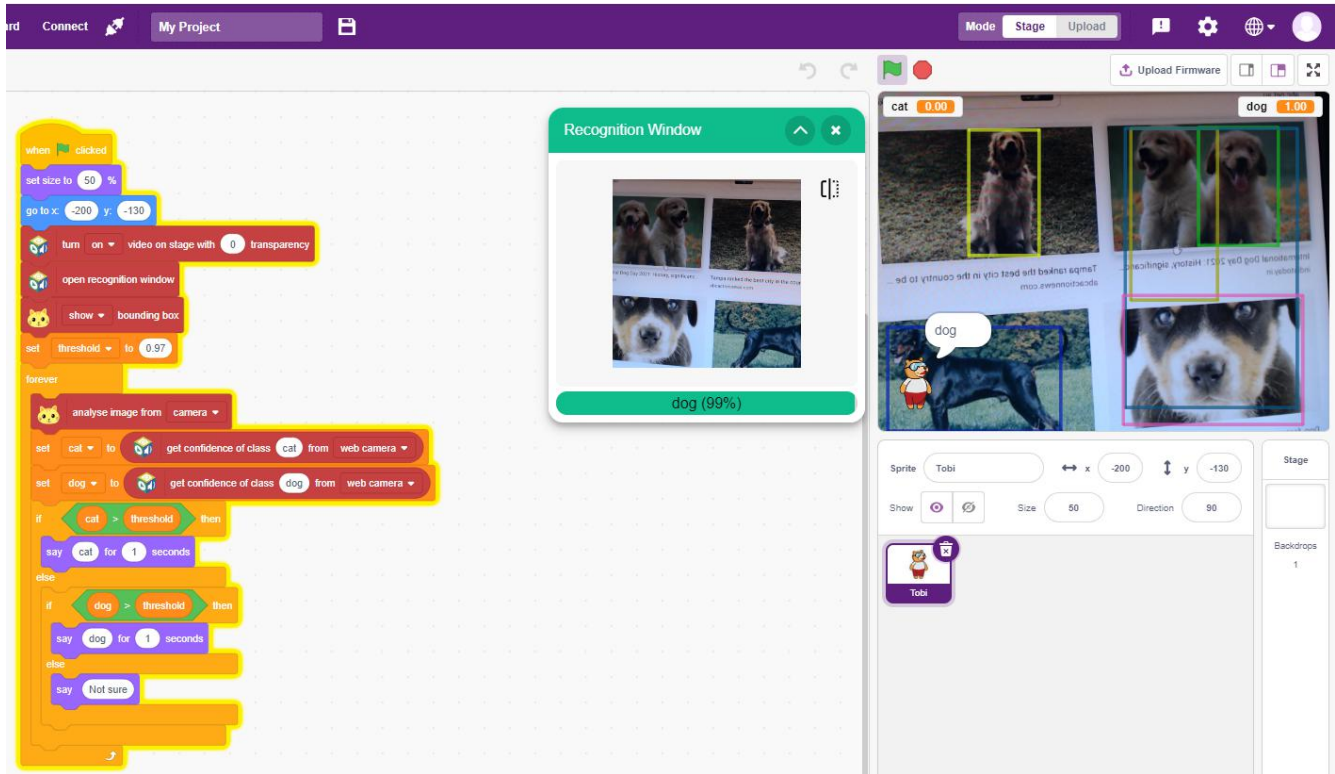
Step 7a: First, insert **set threshold to 0.97** block before forever loop. And now here is the main part in which we first analyse the image **from the camera** and store the confidence **of class** values in variables

```
when clicked
  set size to 50 %
  go to x: -200 y: -130
  turn on video on stage with 0 transparency
  open recognition window
  show bounding box
  forever
    analyse image from camera
    set cat to get confidence of class cat from web camera
    set dog to get confidence of class dog from web camera
```





Step 9a: Start the program and test it on your cat or dog. If you don't have any near you, Google images are here, just search for dog/cat pictures and point your web camera to screen with results. You will see the bounding boxes around detected objects. Try to lower or increase the threshold value or try to confuse the detecting algorithm with some specific cat or dog breeds pictures.



So what exactly is object detection?

To answer that question let's start with image classification. In this task we've got an image and we want to assign it to one of many different categories (e.g. car, dog, cat, human,...), so basically we want to answer the question "What is in this picture?". Note that one image has only one category assigned to it. In simple words, object detection is a type of image classification technique, and besides classifying, this technique also identifies the location of the object instances from a large number of predefined categories in natural images.



http://erasmus-artie.eu

CONCLUSION

Object detection is an image classification technique, and in addition to classifying, this technique also identifies the position of an object found from a large number of predefined categories.



Learning scenario 7

Duration: 90 min

Topics

Interpreting outputs of object classification procedure

Aims

Practical use of object classification

Outcomes

Exploring the possibilities of machine learning extension in Scratch

Object classification project

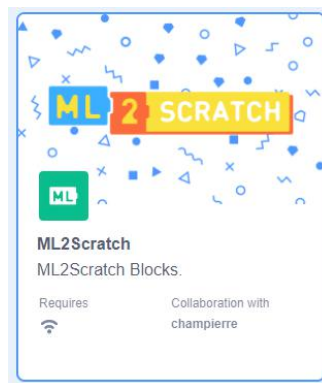
*What is object detection?
How does object detection work?*

Announcement of the goal of today's lesson:
Understanding the process of object classification and its use through a project with practical work.

MAIN PART

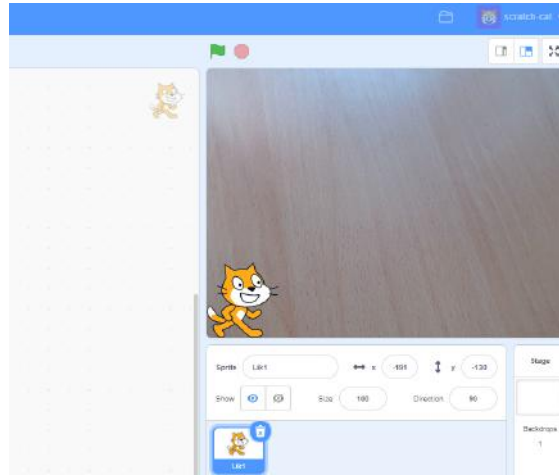
Step 1: Connect webcam

Step 2: Open Scratch at <https://stretch3.github.io/> and Add extension "ML2SCRATCH"

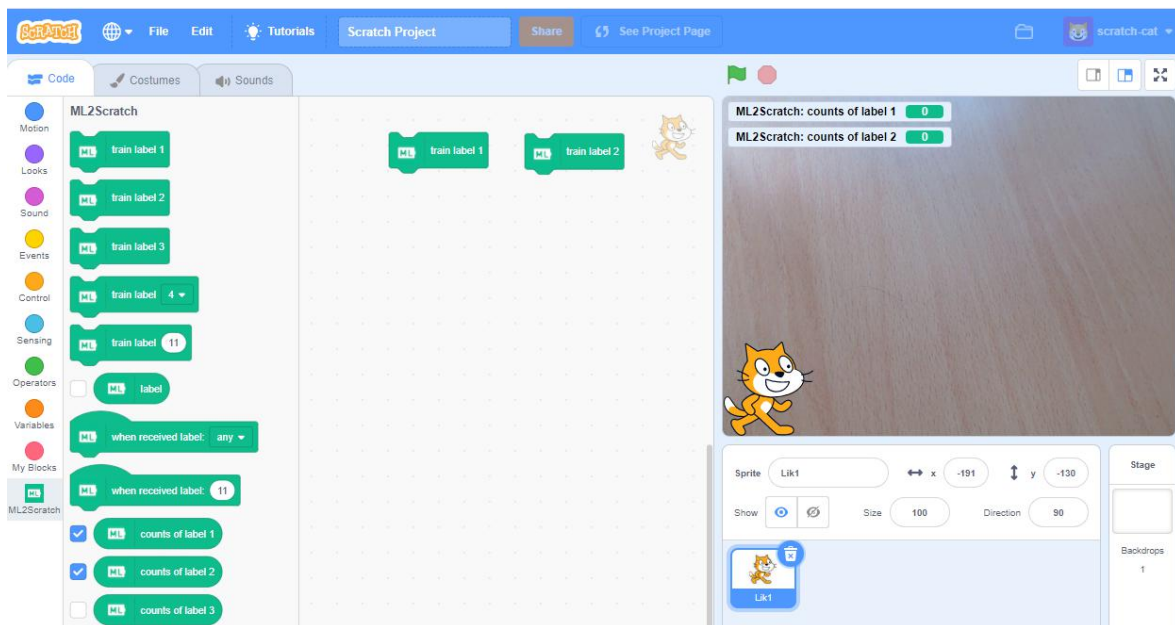




Step 3: Clean up the desk plug in and point webcam to an empty area, move a cat sprite to the corner as shown



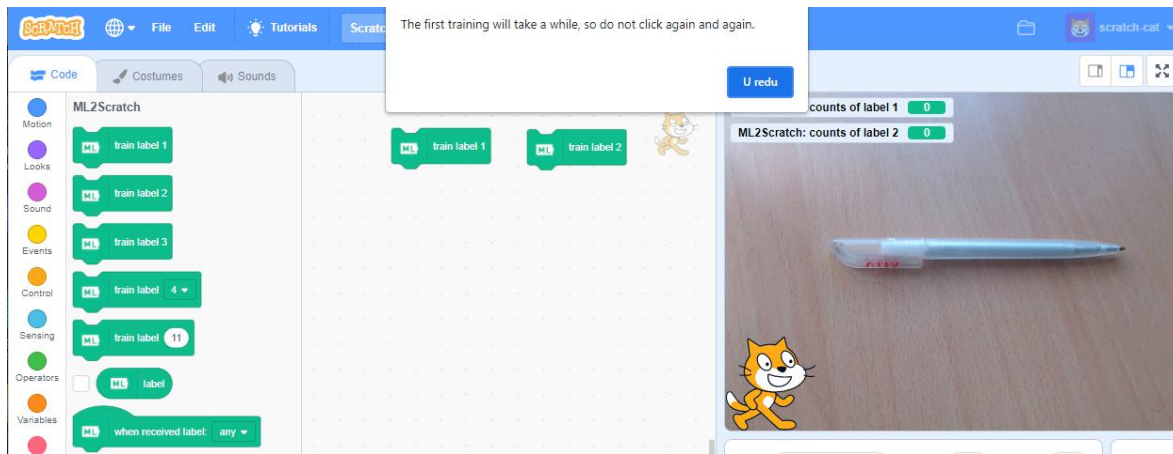
Step 4: From ML2SCRATCH group pick a **train label 1** and **train label 2** blocks and place it on a programming area and check **counts of labels 1** and **counts of labels 2** as shown



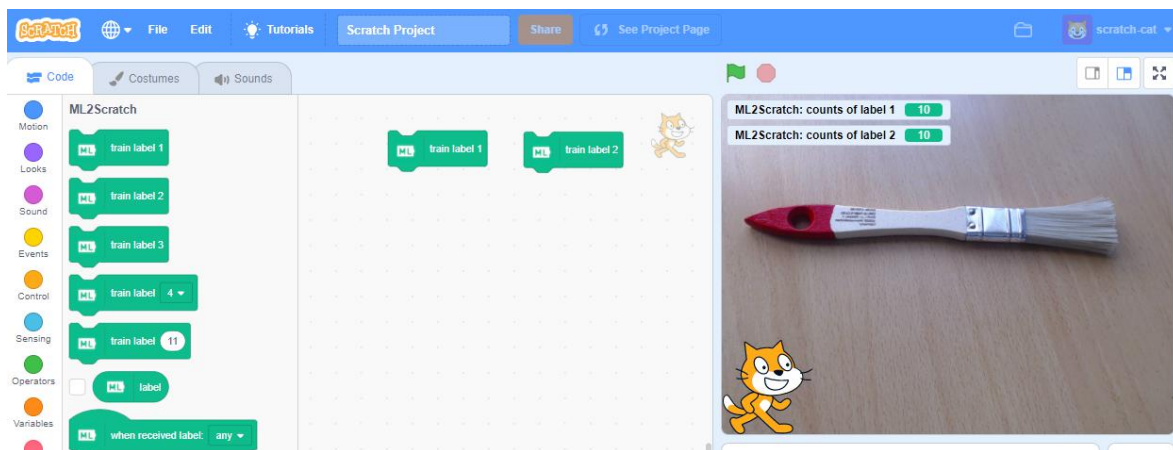


Step 5: Prepare two groups of objects for machine learning. In our case brushes and pens will be used for training the labels

Step 6: Place a first object from first group to the area the webcam is pointed at and click on train label 1 block - you will be notified to wait a bit as shown and label count will change to 1



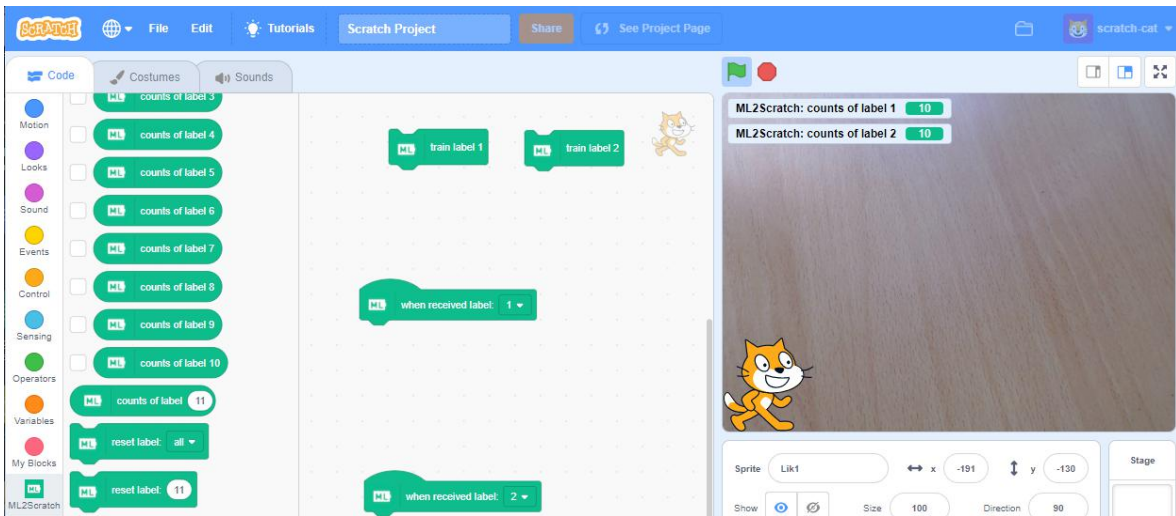
f Step 7: Take approximately 10 images of each object and vary a shape and placement. Do not forget to train the correct label, do not mix objects and labels.



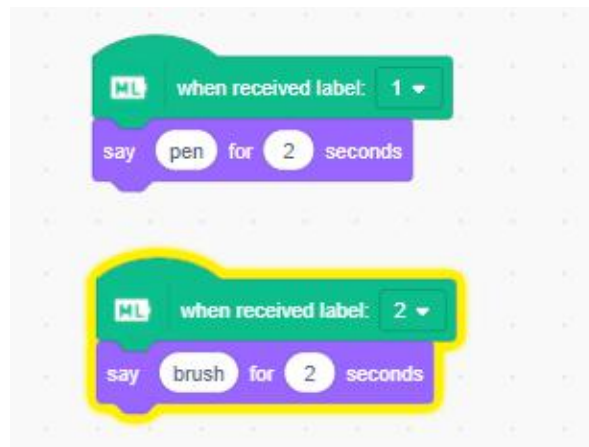
<http://erasmus-artie.eu>



Step 8: Now pick two blocks **when received label: any** and change any to 1 on first and change any to 2 on second block as shown



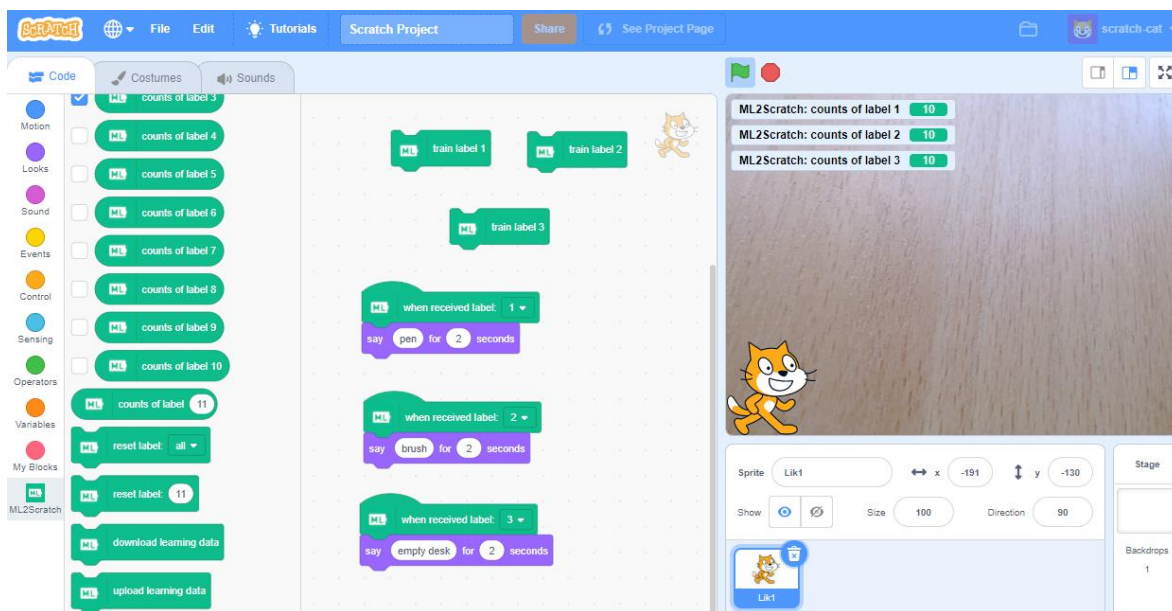
Step 9: From the Look group pick two **say Hello! for 2 seconds** blocks and change “Hello!” to “pen” on the first block and change “Hello!” to “brush” on second block as shown





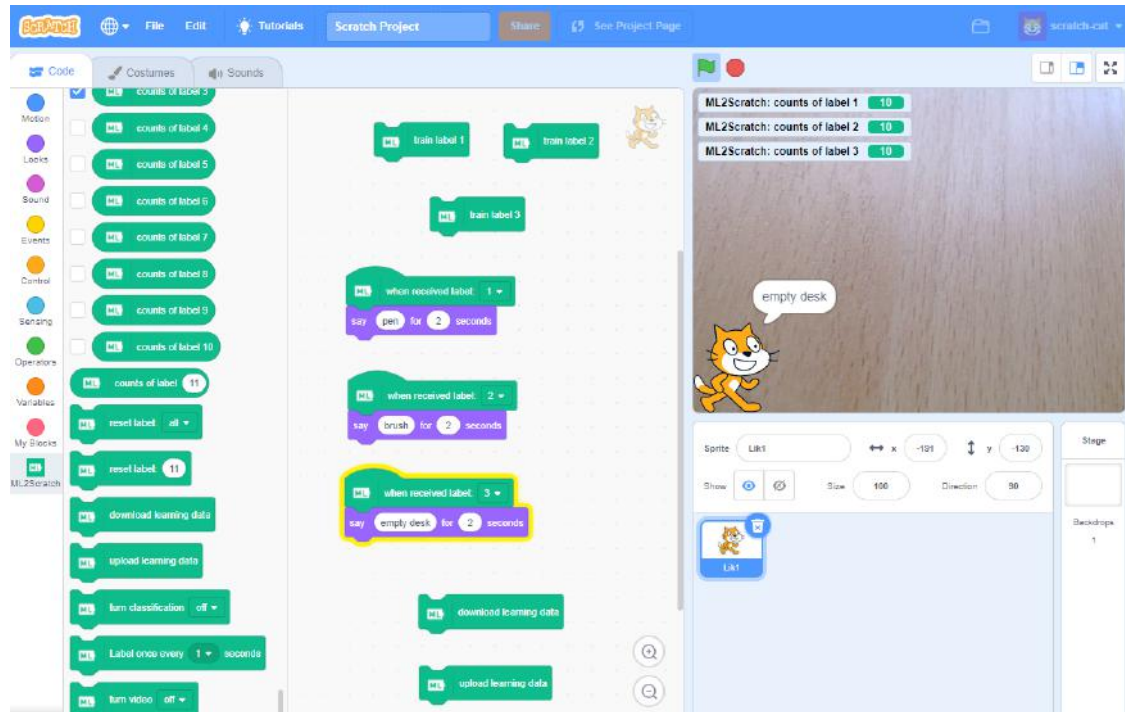
Step 10: Place randomly objects from two groups in the camera view area and note what's happening. Does it work? What's going on when no object is placed? How to solve an empty desk glitch?

Step 11: Solution: Use the third label, train it on empty desk and add the **when received label 3** block with **say empty desk for 2 seconds** block as shown



Step 12: Discuss the accuracy of prediction. Try to train each label more and compare it with previous results. Does the accuracy improve?

Step 13: Use more groups of objects to train more labels. Download and upload your trained data with **download learning data** and **upload learning data** blocks (just click on block to save or upload .json file)



In simple words, image classification is a technique that is used to classify or predict the class of a specific object in an image. The main goal of this technique is to accurately identify the features in an image.

In general, the image classification techniques can be categorized as parametric and non-parametric or supervised and unsupervised as well as hard and soft classifiers. For supervised classification, this technique delivers results based on the decision boundary created which mostly relies on the input and output provided while training the model. But, in the case of unsupervised classification, the technique provides the result based on the analysis of the input dataset on its own; features are not directly fed to the models. The main steps involved in image classification techniques are determining a suitable classification system, feature extraction, selecting good training samples, image pre-processing and selection of appropriate classification method, post-classification processing, and finally assessing the overall accuracy. In this technique, the inputs are usually an image of a specific object, and the outputs are the predicted classes that define and match the input objects. Now that we know all of this, please compare object classification and object recognition.

CONCLUSION

Image classification is a technique used to classify or predict the class of a particular of the object in the picture. The main goal of this technique is to accurately identify features in an image..



Learning scenario 8

Duration: 90 min

Topics

Speech recognition and generation for beginners in Scratch

Aims

To learn about Speech recognition and generation for beginners in Scratch

Outcomes

Understanding Speech recognition and generation with the help of a simple example in Scratch.

Speech recognition and generation for beginners in Scratch

Introduction to Speech Recognition

Speech Recognition is the ability to translate a dictation or spoken word to text. It is also known as Speech-to-Text (STT) and Voice Recognition.

It is achieved by following certain steps and the software responsible for it is known as a „Speech Recognition System“. SR systems are usually implemented in the form of dictation software and intelligent assistants in personal computers, smartphones, web browsers and many other devices.

Introduction to Speech Generation

Speech generation or synthesis (also abbreviated as TTS, Text-to-Speech), unlike speech recognition, is not a technology that exploits the voice, it produces it. Synthetic voices are generally the final phase of the process and are becoming increasingly democratic because they are important in the overall experience of “voice”. Speech synthesis (TTS) is defined as the artificial production of human voices. The main use (and what induced its creation) is the ability to translate a text into spoken speech automatically.

Announcement of the goal of today's lesson

An introduction to speech recognition and generation for beginners through a simple example program in the Makeblock application.



<http://erasmus-artie.eu>



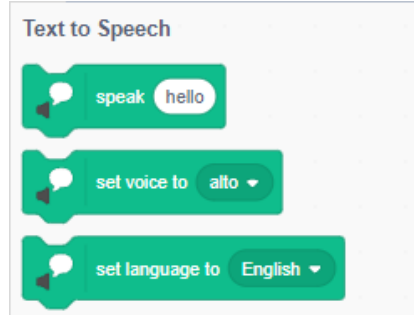
MAIN PART

Applications

Scratch (MIT)

<https://scratch.mit.edu/projects/editor/>

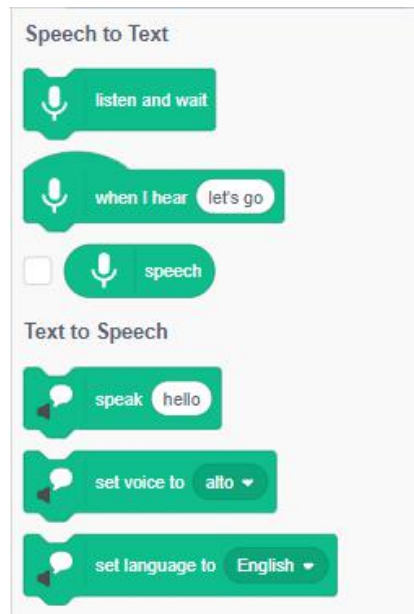
Only Text-to-Speech extension is available (3 blocks)



Scratch (ML4KIDS)

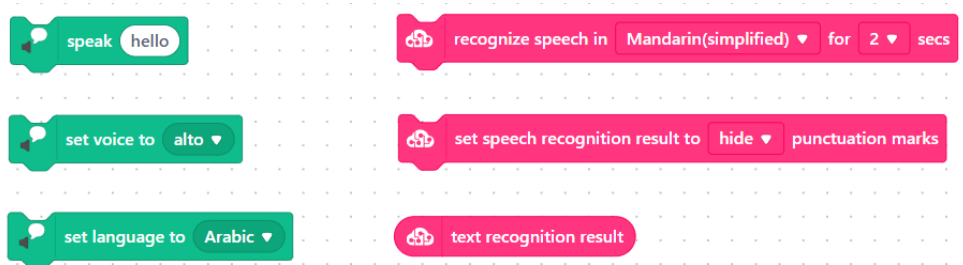
<https://machinelearningforkids.co.uk/scratch3/>

Text-to-Speech (3 blocks) and Speech-to-Text (3 blocks) extensions are available



Makeblock - <https://ide.mblock.cc/>

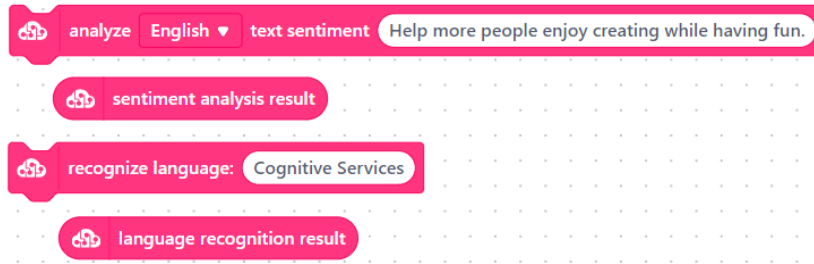
Text-to-Speech (3 blocks), Cognitive services (3 blocks related to Speech-to-Text)





**BONUS*

Makeblock also includes blocks for language recognition and text sentiment analysis



Sentiment analysis is the process of detecting positive or negative sentiment in text. It's often used by businesses to detect sentiment in social data, gauge brand reputation, and understand customers. Sentiment analysis models focus on polarity (positive, negative, neutral) but also on feelings and emotions (angry, happy, sad, etc), urgency (urgent, not urgent) and even intentions (interested v. not interested).

Learn more about Sentiment analysis: <https://monkeylearn.com/sentiment-analysis/>

To illustrate this, let's take an example with Makeblock.

Step 1: Open Makeblock page: <https://ide.mblock.cc/>

Step 2: Add extensions: Cognitive services and Text to Speech

Step 3: Check to display the following reporter type blocks:

- ✓ **speech recognition result**
- ✓ **language recognition result**
- ✓ **sentiment analysis result**

Step 4: Use **speech recognition result** in **recognize language** and **analyse text sentiment** blocks

Step 5: Use this sequence of blocks





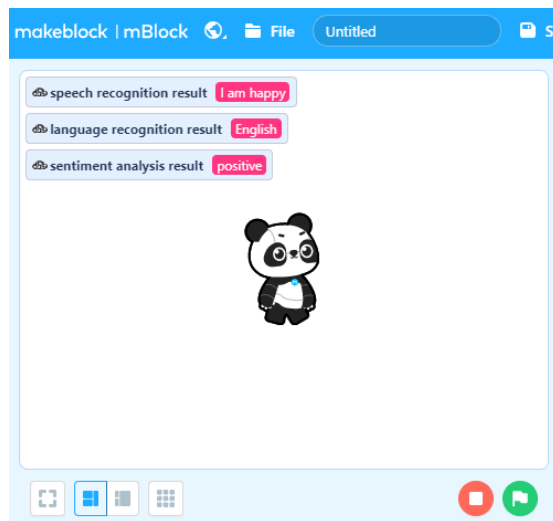
```

when clicked
  recognize speech in English for 2 secs
  recognize language: speech recognition result
  analyze English text sentiment speech recognition result
  wait 2 seconds
  set voice to tenor
  set language to English
  if sentiment analysis result = negative then
    speak This is not good
  if sentiment analysis result = positive then
    speak This is good

```

f

Step 6: Turn on your microphone and speakers, start the code and say "I am happy" (you will see the pop-up window recording your voice for 2 seconds - if you need more, increase the value in **recognize speech in English for 3 secs** (or more). You will hear "This is good" or "This is not good" depending on sentiment analysis results





Step 7: Start the code again, say something else and wait for sentiment analysis results

Basic Principles of Speech Recognition

The smallest unit of spoken language is known as a Phoneme. The English language contains approximately 44 phonemes representing all the vowels and consonants that we use for speech. We can take the example of a typical word such as moon which can be broken down into three phonemes: m, ue, n.

To interpret speech, we must have a way of identifying the components of spoken words and phonemes that act as identifying markers within a speech. An algorithm must be used to interpret the speech further. The Hidden Markov Model is a commonly used mathematical model used to do this. To create a speech recognition engine, a large database of models is created to match each phoneme.

Learn more: <https://www.ibm.com/cloud/learn/speech-recognition>

Basic Principles of Speech Generation

Unlike speech recognition systems that use phonemes (the smallest units of sound) in the first place to cut out sentences, TTS will be based on what are known as graphemes: the letters and groups of letters that transcribe a phoneme. This means that the basic resource is not the sound, but the text. This is usually done in two steps.

f The first will cut the text into sentences and words (our famous graphemes) and assign phonetic transcriptions, the pronunciation, to all these groups. Once the different text/phonetic groups have been identified, the second step consists of converting these linguistic representations into sound. In other words, to read these indications to produce a voice that will read the information.

Try TTS online: <https://www.readspeaker.com/>

Speech recognition is the ability to translate a dictation or spoken word to text. It is also known as Speech-to-Text and Voice recognition. It is achieved by following certain steps and the software responsible for it is known as a “Speech Recognition System”. Speech recognition systems are usually implemented in the form of dictation software and intelligent assistants in personal computers, smartphones, web browsers and many other devices.

CONCLUSION

Speech recognition is the ability to translate dictation or the spoken word into text. Speech generation is a technology that creates a voice.



Learning scenario 9

Duration: 90 min

Topics

Programming speech recognition in Scratch

Aims

To learn to program speech recognition with uploaded examples

Outcomes

Knowing how to write a program for speech generation using Scratch

Programing speech recognition in scratch

Ask students what is speech recognition?

Ask students what is SIRI and Google NOW?

Do they know any other Speech Recognition System?

In recent years, speech recognition technology has grown increasingly widespread. This technology is frequently employed by companies and individuals alike because of the numerous benefits it brings. “Hey, Siri,” “OK Google,” and so on – Voice recognition, often known as speech recognition technology, is not a new concept (SRT). It refers to a type of technology that can transform spoken words into machine-readable forms. You can now communicate to your devices and have them act on your commands, very much like in science fiction tales. Because most speech recognition technologies have an accuracy rate of over 95%, it’s no wonder that latest voice search statistics reveal almost 50% of all searches in 2022 are done speaking.

Announcement of the goal of the lesson:

Introduction to the speech recognition program and its use on the example of one program.



<http://erasmus-artie.eu>



MAIN PART

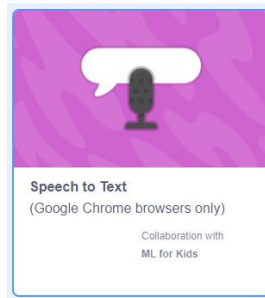
If we want to work with speech recognition, extensions are available in Scratch and Makeblock applications.

SCRATCH (ML4KIDS):

Step 1: Open your Chrome web browser and go to:

<https://machinelearningforkids.co.uk/scratch3/>

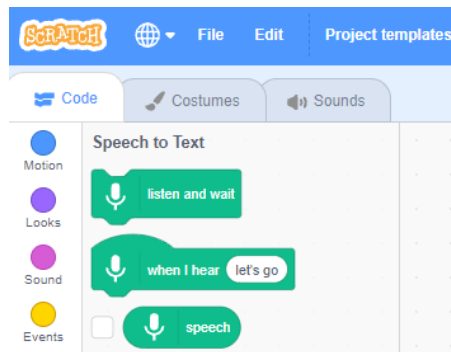
Step 2: Load extension Speech to Text



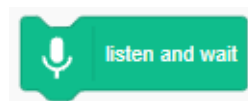
Step 3: You will see the new group in Block Palette called „ Speech to Text “and 3 new blocks there



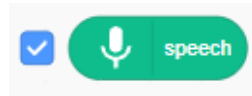
http://erasmus-artie.eu



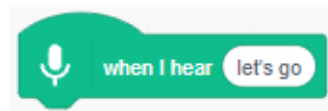
Step 4: Listen and wait block starts listening and processing of spoken words



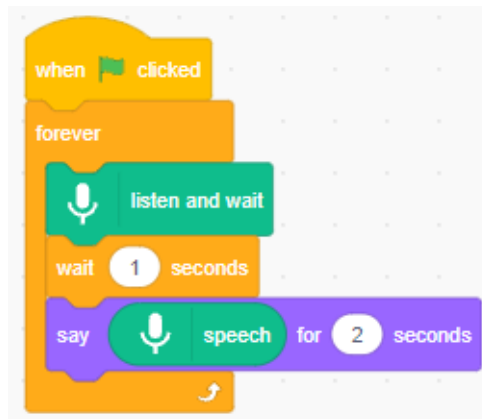
Step 5: Results of speech recognition are displayed in the speech reporter block. Check if you want the speech recognition result to be displayed on stage .



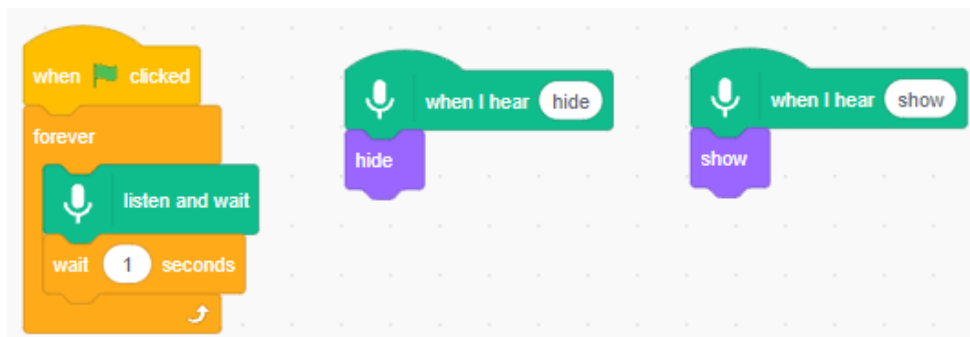
Step 6: The last block is the event trigger block. This block waits for the word in white balloon, such as „something“ in this example and then executes the sequence of blocks attached to it.



Step 7: So, let's make a simple listen and say program. All you have to do is place, **listen and wait** in a loop with **say** block to see the speech recognition in action.



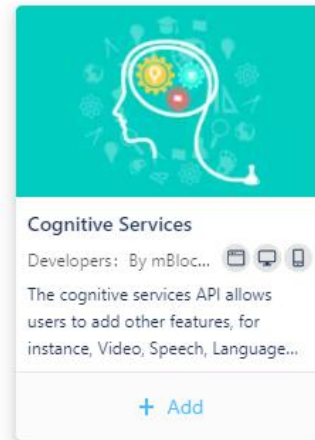
Step 8: You can also wait for a specific word(s) to trigger the event(s). Such as this hide and show game.



MAKEBLOCK

Step 1a: Open your web browser and go to:
<https://ide.mblock.cc/>

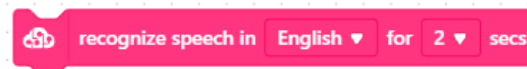
Step 2a: Load Sprite extension Cognitive Services



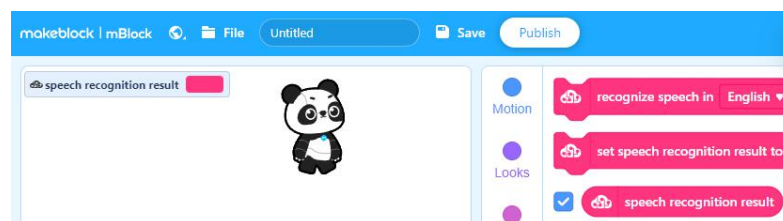
Step 3a: There are many blocks in this group but only few are related to speech recognition.



Step 4a: **Recognize speech in <language> for <x> seconds** block starts speech recognition for several seconds. RECOGNITION pop-up window will appear, and you will see a waveform when speaking.



Step 5a: Results of speech recognition are displayed in the **speech recognition result** block. Check if you want the speech recognition result to be displayed on stage.



Step 6a: the last block will display or hide the punctuation marks from speech recognition result



```

set speech recognition result to hide punctuation marks
display
  ✓ hide

```

Step 7a: So, let's make a simple listen and say program in Makeblock. You will see it's very similar to the Scratch version.

```

when clicked
  forever
    recognize speech in English for 2 secs
    wait 1 seconds
    say speech recognition result for 2 seconds

```

Speech recognition technology is now a part of our daily life, although it is still confined to simple requests. Researchers will be able to develop more sophisticated systems that interpret conversational speech as technology progresses. One day, you'll be able to converse with your computer in the same manner that you would with a human, and it would respond with rational replies. Signal processing technology will make all of this feasible. The demand for professionals in this sector is increasing, and many organisations are searching for bright people to join them. Many powerful new technologies and ways of communication rely on processing, interpreting, and comprehending voice signals. Considering current trends, speech recognition will continue to be a fast-growing subset of signal processing in the following years.

CONCLUSION

Speech recognition technology enables voice communication between users and computers.



<http://erasmus-artie.eu>



Learning scenario 10

Duration: 90 min

Topics

Programing speech generation in Scratch

Aims

To learn to program speech generation with uploaded examples

Outcomes

Knowing how to write a program for speech generation using Scratch

Programing speech generation in scratch

Text-to-speech (TTS) is a type of assistive technology that reads digital text aloud. It's sometimes called "read aloud" technology. With a click of a button or a touch of a finger, TTS can take words on a computer or other digital device and convert them into audio. TTS is very helpful for kids and adults who struggle with reading. But it can also help with writing and editing, and even with focusing.

A Concept-to-Speech (CTS) system converts the conceptual representation of a sentence-to-be-spoken into speech. While some CTS systems consist of independently built text generation and Text-to-Speech (TTS) modules, most of the existing CTS systems enhance the connection between these two modules with a prosodic prediction module that utilizes linguistic knowledge from the text generator to predict prosodic features for TTS generation.

Speech generation can transform any text into speech. Speech generation is producing spoken messages in response to signals from a data processing or control system. The selection of messages is produced by assembling speech sounds from a set of fundamentals that may be artificial in origin or may have been extracted by processing those human-produced.

Announcement of the objective of the lesson:

An introduction to the speech generator and its use through an example program.



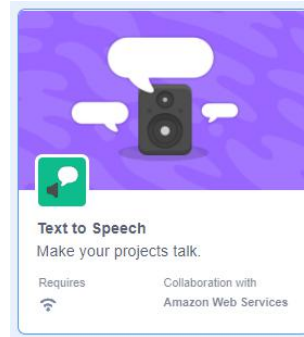
<http://erasmus-artie.eu>



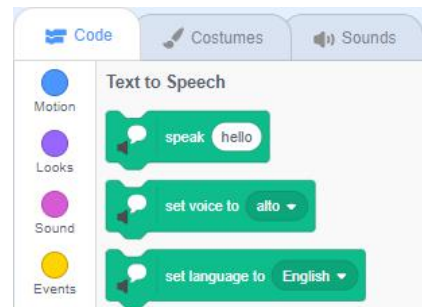
MAIN PART

Step 1: Open your Chrome web browser and go to: <https://machinelearningforkids.co.uk/scratch3/>

Step 2: Load extension Text to Speech

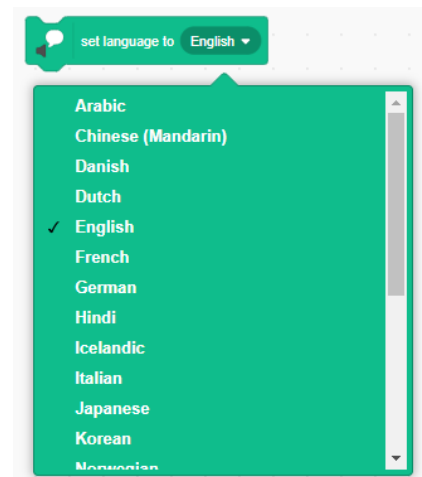


Step 3: You will see the new group in Block Palette called „Text to Speech “and 3 new blocks there



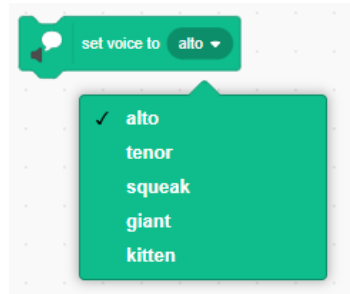
Step 4: Let's go from the bottom – **set Language to block**

This block sets the output language – you can choose it from the dropdown list

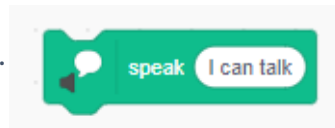




Step 5: Next block - **set voice to** set the voice type. You can choose: alto, tenor, squeak, giant or kitten.



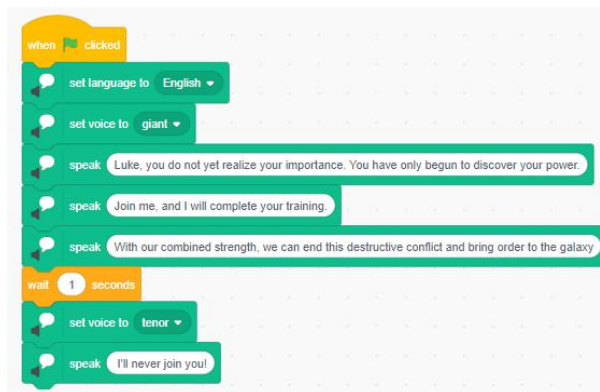
Step 6: And the most important block - **speak** block. This block “speaks” the text in white balloon, such as „I can talk“ in the example below. Change this to anything you want and click on the block to hear it. Ensure your speaker volume is turned on before testing.



Step 7: Basically, not the hardest work to make the Scratch sprite talk. All you must do is to set the language, voice and start talking.

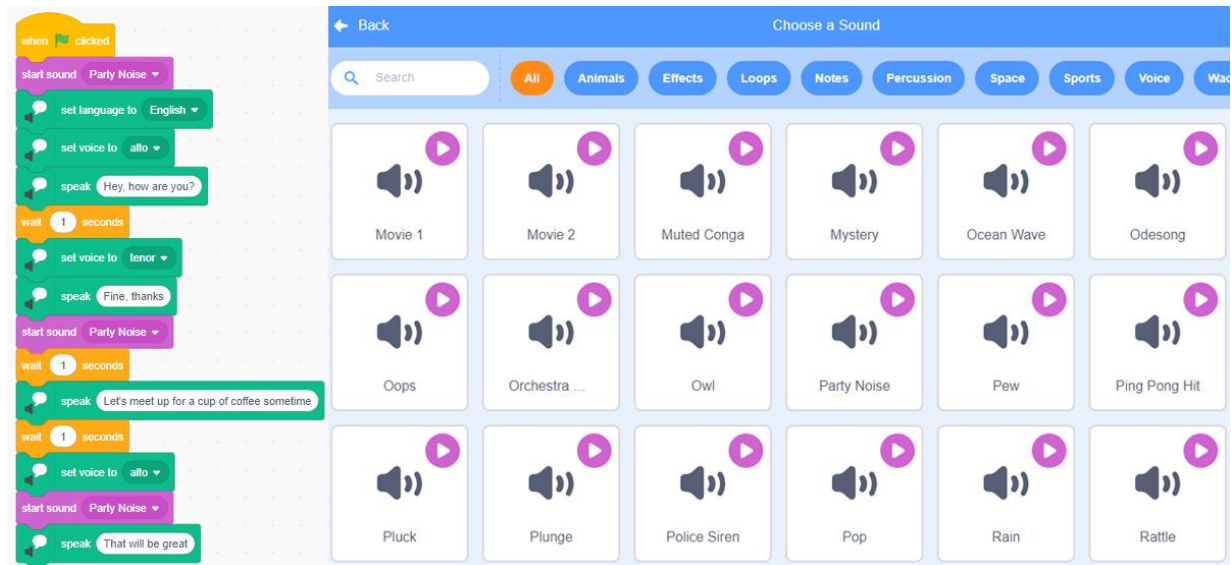


Step 8: What can you do with it? You can recreate the famous movie lines, like this one from Star Wars



Step 9: Or make your own movie, tell a story... Combine it with other clips from sound gallery to make the scene more realistic.





Today we have many speech generation devices. Speech-generating devices let people ‘speak’ words and sentences electronically. Speech-generating devices are hand-held electronic devices that play words or phrases when the user touches a switch or presses buttons or keys. Some devices ‘speak’ words as the words are typed on a keyboard. Speech-generating devices allow people who can’t use spoken language to ‘speak’ electronically. Speech-generating devices have been used to help autistic children communicate since the 1990s.



CONCLUSION

Converting text to speech (Text To Speech - TTS) is a type of technology that reads digital text aloud.



Learning scenario 11

Duration: 90 min

Topics

Interpreting outputs of voice-controlled object algorithm

Aims

Practical use of voice-controlled object

Outcomes

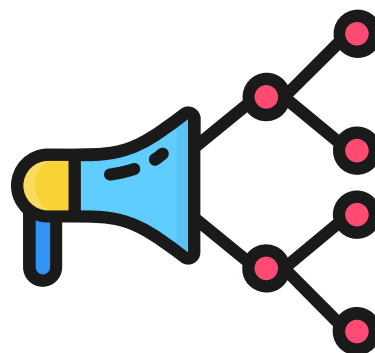
Exploring the possibilities of Speech to Text extension in Scratch

Project with voice-controlled object

*Ask your students if an object can be voice controlled.
Is it possible to benefit from voice-controlled objects in some way?*

Announcement of the goal of the lesson:

Understanding the voice control algorithm of the facility and its use in practical work.



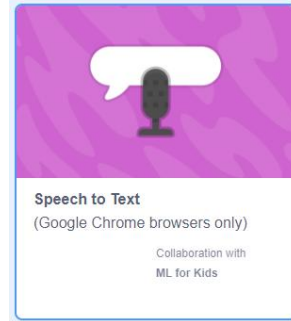


MAIN PART

Through a project, the teacher instructs students on Scratch commands and skills as well as on how to train a model to convert speech to text.

Step 1: Open your Chrome web browser and go to:
<https://machinelearningforkids.co.uk/scratch3/>

Step 2: Load extension Speech to text
(STT - Google Chrome browsers only)

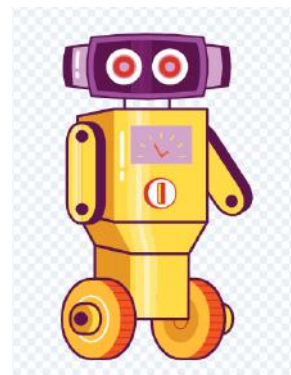


Step 3: Delete Cat sprite by clicking on sprite trashcan icon

Step 4: Download maze.png from:
<https://drive.google.com/file/d/11YBBhQclhVfHYMWeLkhgAYKSsvfv33pT5/view?usp=sharing> and upload it to Scratch as custom sprite



Step 5: From sprite gallery choose **Retro Robot** and use the second costume (Retro Robot b)



Step 6: From sprite gallery choose **Home Button**



<http://erasmus-artie.eu>



Step 7: Make 3 variables (for all sprites):

- **gameover** (shows how the game ended)
- **xm** (x position of robot)
- **ym** (y position of robot)

Step 8: Make a list and change its name to: **endtalk** (displays the game over message, depending on how the game ended)

Step 9: Switch to **maze sprite** and start coding. First few blocks are setting the position, size, and visibility of the maze. Forever loop that follows is a collision checker. In case of collision, the **gameover** variable is set to 1.

Step 10: Switch to **Home Button** sprite. The code is almost the same except that in case of collision a **pop** sound is played and the **game over** variable is set to 2.

```
when green flag clicked
  go to x: 0 y: 0
  set size to 120 %
  show
  forever loop
    if touching Retro Robot ? then
      set gameover to 1
```

```
when green flag clicked
  set size to 100 %
  go to x: 195 y: -120
  show
  forever loop
    if touching Retro Robot ? then
      play sound pop until done
      set gameover to 2
```



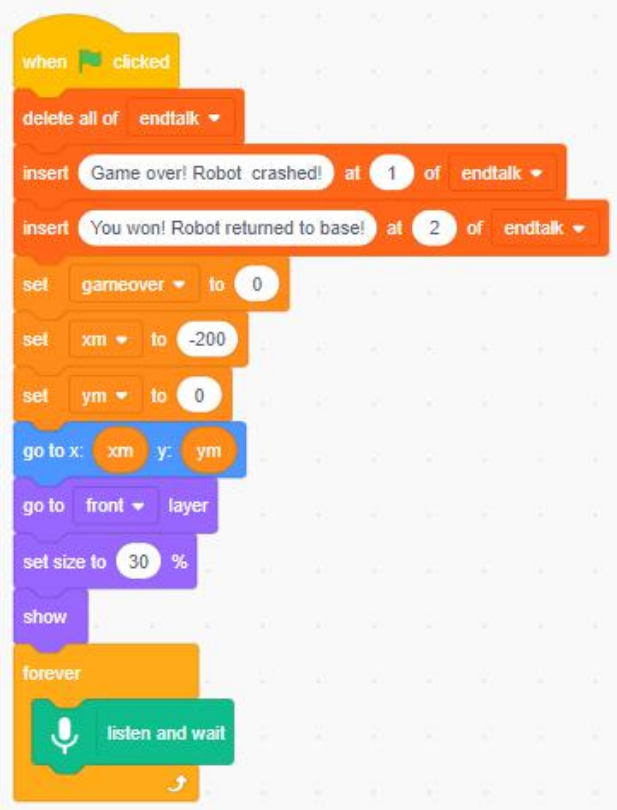


Step 11: The main code is assigned to **Retro robot** code and there are 5 threads which start simultaneously.

11.1 Flag (start) code sets the variables, list entries (and deleting all previous entries), position and visibility. At the end there is a listening block (from STT extension) in the loop.

11.2 The next 4 threads are triggered by speech recognition and each recognized voice command is handled by a sequence of blocks. If no collision is detected (gameover is 0) robot moves in specific direction (by changing xm or ym value and moving robot to calculated position).

If collision is detected (gameover is 1 or 2) game over message (from the list) will be displayed and the program will stop.





```
when I hear up
if (gameover = 0) then
  change ym by 20
  glide 0.25 secs to x: xm y: ym
else
  start sound computer beeps1
  say item gameover of endtalk for 3 seconds
  stop all

when I hear left
if (gameover = 0) then
  change xm by -20
  glide 0.25 secs to x: xm y: ym
else
  start sound computer beeps1
  say item gameover of endtalk for 3 seconds
  stop all

when I hear right
if (gameover = 0) then
  change xm by 20
  glide 0.25 secs to x: xm y: ym
else
  start sound computer beeps1
  say item gameover of endtalk for 3 seconds
  stop all

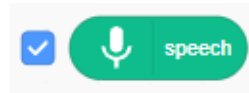
when I hear down
if (gameover = 0) then
  change ym by -20
  glide 0.25 secs to x: xm y: ym
else
  start sound computer beeps1
  say item gameover of endtalk for 3 seconds
  stop all
```





Step 12: Try to use your native language voice commands and see how it works.

Discuss how (in)correct spelling affects the speech recognition.
Turn on (check) speech recognition results.



Object tracking robots, if they can be controlled smartly through voice, can be of tremendous help for the physically handicapped people. A speech recognition system is used to recognize a set of predefined commands such as forward, backward, left, right and rotation at a particular turning angle. The robot navigates its way as per the voice-command signal, while also tracking the desired object. The voice-command signal processing is carried out in real-time, using an on-time cloud server that converts it to text form. The command signal text is then transferred to the robot via Bluetooth network to control its differential drive. The prototype smart robot consists of three sub-systems: speech recognition system, object tracking system and the differential-drive based movement's control system. The accuracy and efficiency of the speech recognition system is examined through a set of experiments. The effect of factors such as noise and distance etc. are examined, with encouraging results. The prototype robot can recognize the voice-commands within a Bluetooth range, i.e, 10 m. Possible extensions are also discussed that could lead to a wide range of further applications.



CONCLUSION

We can control objects in a computer program with our voice.



Learning scenario 12

Duration: 90 min

Topics

Introduction to hardware - microcontroller, camera and motor driver

Aims

Students learn about the hardware to be used on our main project

Outcomes

Becoming familiar with the hardware used on main project

Introduction to hardware - microcontroller, camera and motor driver

A part of this curriculum is related to a real, physical devices with AI capabilities. For that purpose, Croatian Robotic Association developed a small mobile robot to implement programing knowledge from previous lessons. First, let's consider what our robot should consist of. It's clear that mechanical parts - case, wheels and DC motors - will be used as a drive. What about electronic parts? Pretty much like most of the living beings, our robot will need the brain (microcontroller), the eyes (AI camera) and the nerves/impulses (DC motor driver). Let's take a look at those parts.

There are 2 options for building a robot:

Maqueen Plus robot kit + HuskyLens AI camera (easy - suitable for beginners)

micro: Maqueen Plus is an advanced STEM educational robot for Micro:bit. Powerful and smart, this micro:bit robot has optimized power management and a high-capacity power supply: it can be fully compatible with the AI HuskyLens vision sensor, making it an accessible teaching tool.

Micro:Maqueen Plus features a large and stable chassis, built-in features and multiple expansion ports. It is not only suitable for classroom teaching, but can also be used for extended after-school training sessions and robot competitions.

Arduino UNO + Motor driver + HuskyLens AI camera (more complex - only for experienced users)



<http://erasmus-artie.eu>

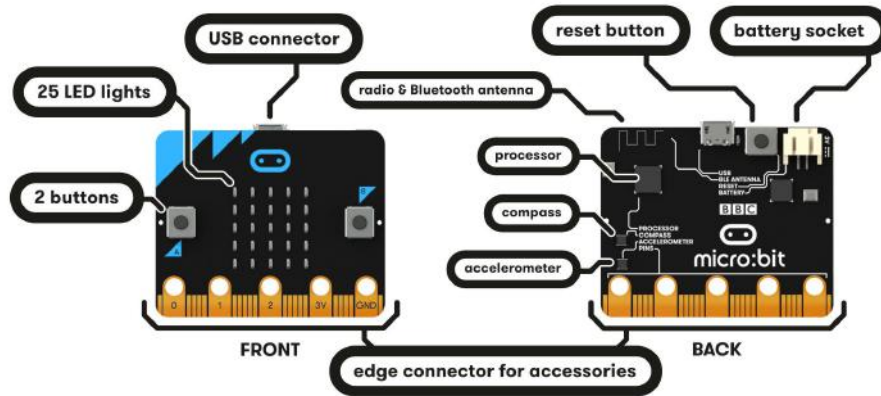


MAIN PART

Brain 1 - micro:bit

A “brain” in electronic world is called a processor or, in this specific case - microcontroller.

Micro:bit is an easy-to-use, powerful and cost-effective pocket-sized microcontroller designed for teaching kids and beginners how to program, allowing them to easily bring their ideas into DIY digital games, interactive projects and robotics.

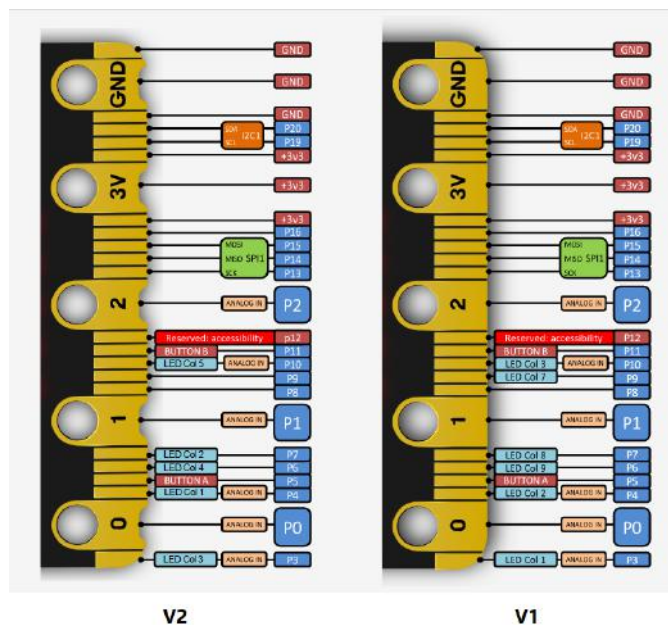


Thanks to its external I/O ports and hardware support, Micro:bit is well suited for various robot-related learning and development.

Micro:bit layout and tech specifications

There are two versions of micro:bit available on market - check this article to find out which one you have:

<https://kitronik.co.uk/blogs/resources/explore-micro-bit-v1-microbit-v2-differences>





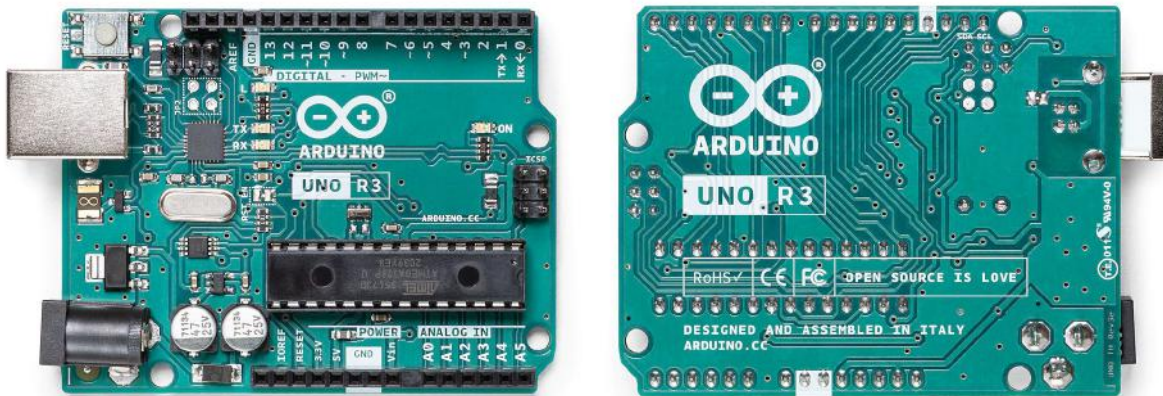
- A small board similar in size to a credit card (4cm x 5cm)
- On-board modules, like accelerometer, Compass and Bluetooth® Smart module
- A pocket-sized microcontroller
- A 5x5 LED matrix (also supports light detection)
- Light and temperature sensors and other common sensing devices

Equipped with ARM’s M0 processor, micro:bit can execute most of the fundamental functions of a robot.

Introduction to micro:bit https://www.youtube.com/watch?v=POkeI_2NXMo

Brain 2 - Arduino UNO

Arduino UNO is the most popular microcontroller in the world with huge online community and plenty of projects available on the internet. There are many UNO clones on the market. A clone means that core architecture, in terms of electronics, is similar to Arduino UNO, but some modifications are made to provide the additional features to the board. These modifications are designed and developed specially for students to learn coding and microcontroller architecture. Let’s take a look at Arduino UNO.



Each of the 14 digital pins on the Uno can be used as an input or output and they operate at 5 volts. Each pin can provide or receive 20 mA as recommended operating condition and has an internal pull-up resistor (disconnected by default) of 20-50k ohm. A maximum of 40mA is the value that must not be exceeded on any I/O pin to avoid permanent damage to the microcontroller.

In addition, some pins have specialized functions:

- Serial: 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
- External Interrupts: 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value.
- PWM: 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output function.

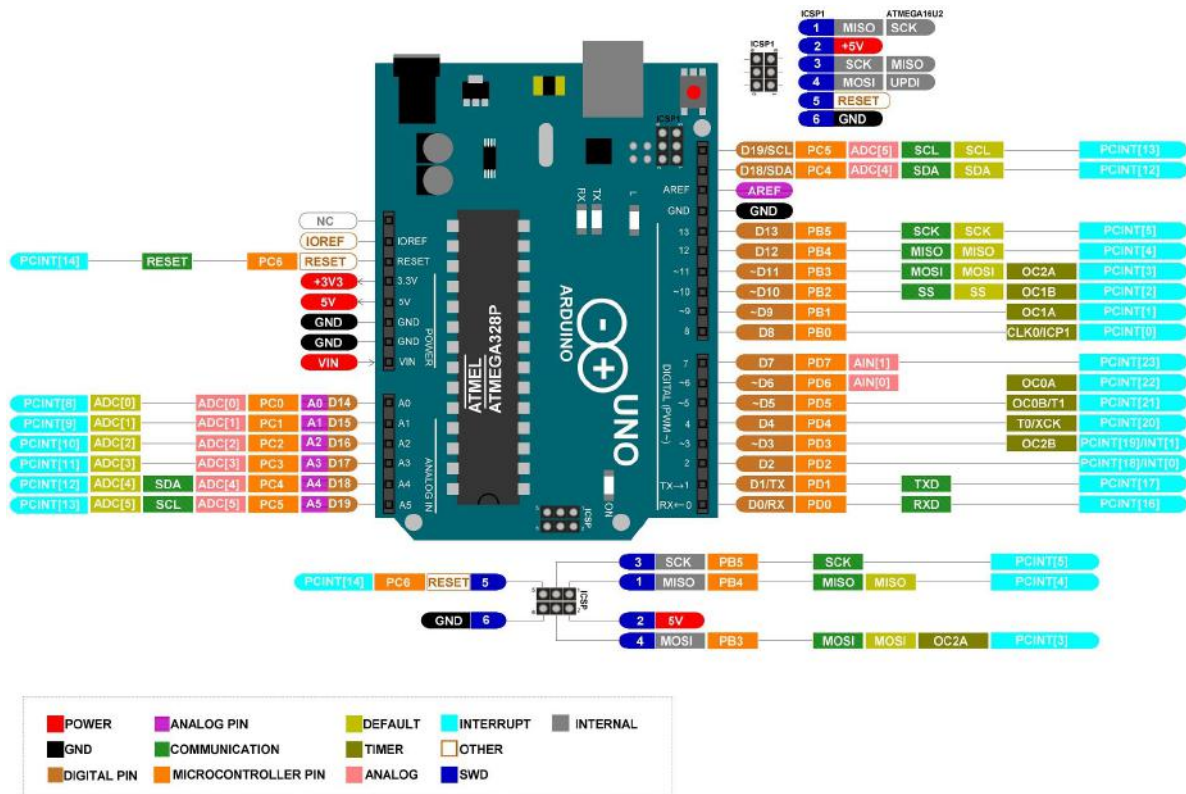


- SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication using the SPI library.
 - LED: 13. There is a built-in LED driven by digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.
 - TWI: A4 or SDA pin and A5 or SCL pin. Support TWI communication using the Wire library.
- The Uno has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default, they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin. There are a couple of other pins on the board:

- AREF. Reference voltage for the analog inputs.
- Reset. Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board

Arduino UNO layout and tech specifications

Arduino UNO board with pins explained:





MICROCONTROLLER	ATmega328P
OPERATING VOLTAGE	5V
INPUT VOLTAGE (RECOMMENDED)	7-12V
INPUT VOLTAGE (LIMIT)	6-20V
DIGITAL I/O PINS	14 (of which 6 provide PWM output)
PWM DIGITAL I/O PINS	6
ANALOG INPUT PINS	6
DC CURRENT PER I/O PIN	20 mA
DC CURRENT FOR 3.3V PIN	50 mA
FLASH MEMORY	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
CLOCK SPEED	16 MHz
LED_BUILTIN	13
LENGTH	68.6 mm
WIDTH	53.4 mm
-----	--

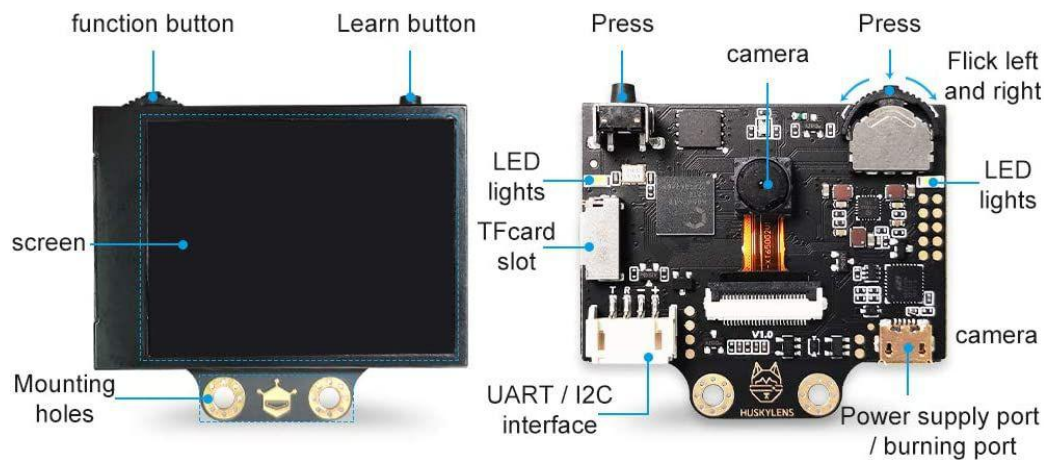
Main power for Arduino UNO is through USB connection but it could be powered from batteries connected via pin header (VIN and GND pins).

Arduino UNO introduction: <https://www.youtube.com/watch?v=bniUECtJkeU>

Eyes - HuskyLens Kendryte K210 (all scenarios)

Our mobile robot will be able to sense the environment with this camera. HuskyLens is an easy-to-use AI computer vision sensor with 7 built-in functions: face recognition, object tracking, object recognition, line tracking, colour recognition, tag recognition and object classification.

It can be easily connected to any Arduino/Arduino compatible device and micro:bit. Now, you can do very creative projects even without the knowledge of complex machine learning algorithms. Let's take a look at the camera.

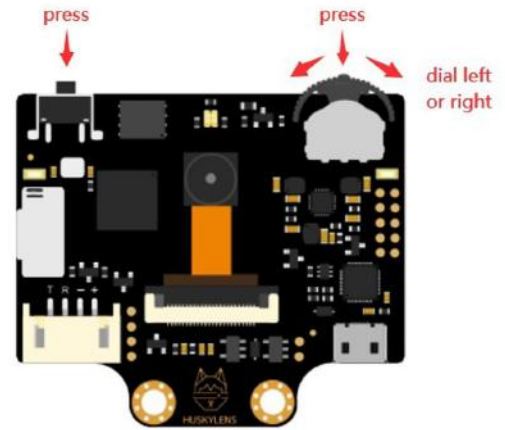




Buttons

There are two buttons on the HuskyLens, the function button and the learning button. The basic operations of these two buttons are shown as follows:

- Dial the "function button" to the left or right to switch different functions.
- Short press the "Learning button" to learn the specified object; long press the "Learning button" to continuously learn the specified object from different angles and distances; if HuskyLens has learned the object before, short press the "Learn button" to make it forget.
- Long press the "function button" to enter the second-level menu (parameter setting) in the current function. Dial left, right or short press the "function button" to set related parameters.

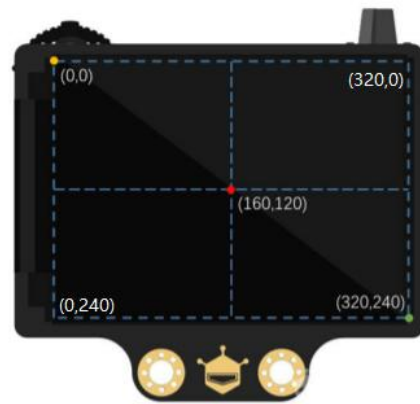


Coordinate system

When HuskyLens detects an object, the target will be automatically selected by a colour frame on the screen. The coordinates of the color frame position x and y are assigned according to the following coordinate system. After getting the coordinates from the UART / I2C port, you can know the position of the object. Format:x,y)

Functions:

- 1) Face recognition
- 2) Object tracking
- 3) Object recognition
- 4) Line tracking
- 5) Colour recognition
- 6) Tag recognition
- 7) Object classification



Colour Instructions

In each function, the colour definitions of the frame and the symbol "+" in the centre of the screen are all the same, which helps you know the current status of HuskyLens.

Colour	Status
From orange to yellow, then from yellow to orange	Have not learned the object yet but ready to learn
Yellow	Learning the new object
Blue	Have learned the object and recognized it





The RGB LED indicator is used to indicate the status of the face recognition function. Its colours are defined as follows.

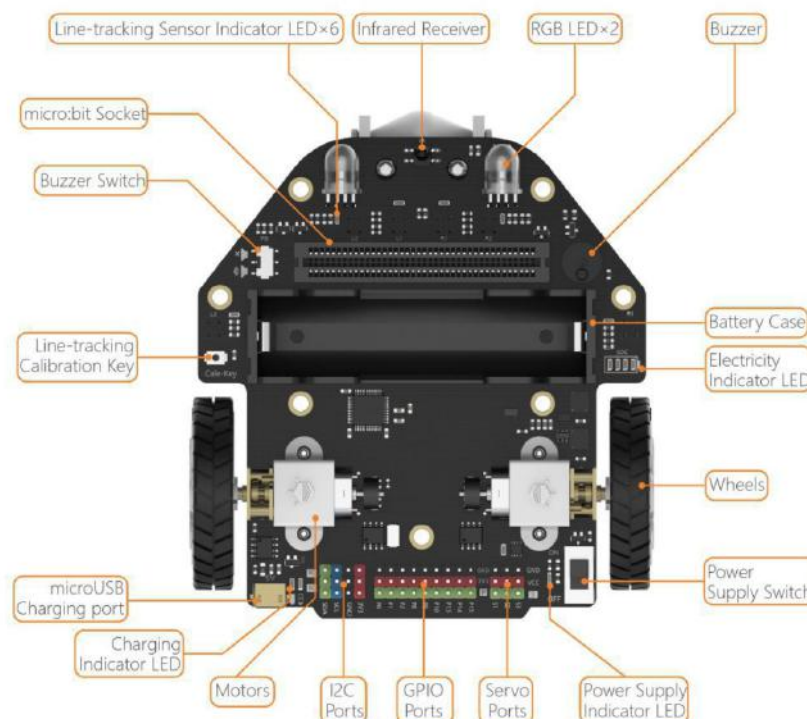
Colour	Status
Blue	Have not learned the face yet, but detected the face
Yellow	Learning the new face
Green	Have learned the face and recognized it

Complete manual: https://wiki.dfrobot.com/HUSKYLENS_V1.0_SKU_SEN0305_SEN0336

Nerves/impulses/muscles 1 - Maqueen Plus robot board

Maqueen Plus is a smart programmable educational robot designed for beginners. It can be programmed with Mind+ and MakeCode programming platforms. It is optimized with better power management and a larger capacity power supply. It's ideal for use with the HuskyLens AI Vision Sensor and comes with a larger and more stable chassis. It also features more built-in functions and more expansion ports. It's suitable for classroom teaching, and for after-school extended exercises and robot competitions.

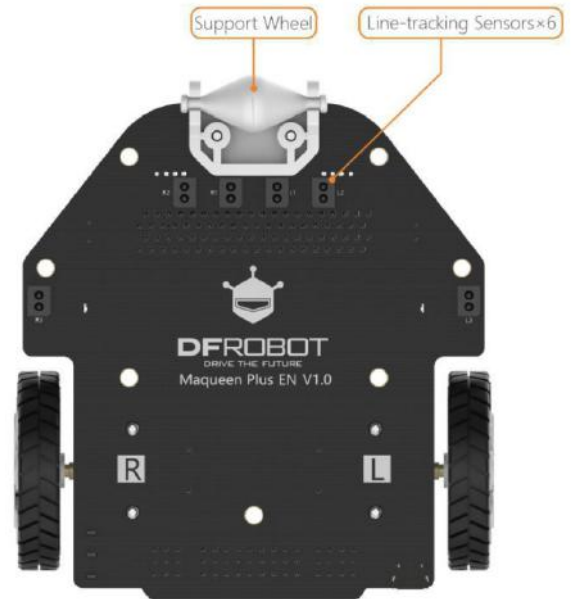
Here are the main parts on the Maqueen Plus board:





Maqueen plus tech specifications

- Power Supply: 3.7V-18650 lithium battery
- Charging Voltage: 5V
- Charging Current: 900mA, Charging Time: 4h
- Battery Indicator: 4 LEDs
- Drive Motor: N20 motor 260 rpm
- Buzzer * 1
- RGB-LED * 2
- GPIO Expansion Ports : P0 P1 P2 P8 P12 P13 P14 P15 P16
- I2C Expansion Ports * 3
- Servo Expansion Ports *3
- Line Tracking Sensors *6
- Line Tracking Sensor Output Data: analog + digital
- Line Sensor Calibration: support
- Infrared Receiving Sensor *1
- Ultrasonic Sensor: URM10
- Top Metal Plate * 1
- M3 threaded connections *12
- Map Size: 50cm*50cm
- Dimension: 107x100mm (4.21 x3.94")



Maqueen Plus is accessible through Mind+ Programming platform now. Mind+ is a Scratch3.0-based graphical programming platform from DFRobot, supporting python, Arduino and other programming platforms. At present, Mind+ has been applied to all kinds of sensors, modules and related educational products.

Find out more about Maqueen Plus:
https://github.com/DFRobot/Maqueen_Plus_Basic_Tutorial/blob/master/MBT0021-EN-Maqueen%20Plus%20Basic%20Tutorial.pdf

As this scenario is being completed, the V2 version of this robot is available on the market.

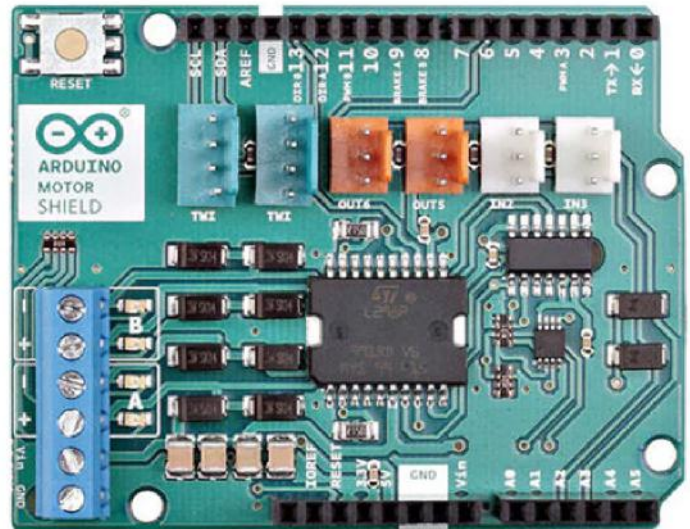




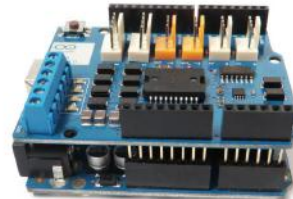
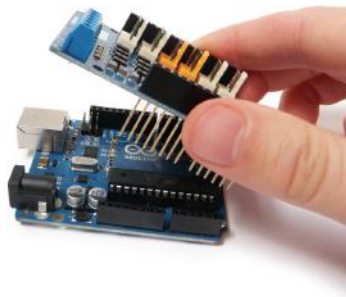
Nerves/impulses/muscles 2 - Arduino Motor Shield Rev 3

There are several ways we can control a DC motor; perhaps the easiest one is just by applying power to it. Very early inventions using the DC motor worked like that: add a power source and the motor will start rotating, switch the polarity and you switch the direction.

But if we want to do a bit more than just making a motor spin full speed in two directions, we need a motor control circuit. More specifically, the dual full-bridge driver L298P (chip), which we can find on the Motor Shield Rev3.



Motor shield is stackable to Arduino UNO and that means you don't have to wire the connection to the microcontroller. Just plug the shield pins (male) to microcontroller pins (female).



The motor shield has 2 channels, which allows for the control of two DC motors, or 1 stepper motor.

It also has 6 headers for the attachment of Tinkerkit inputs, outputs, and communication lines. The use of these pins is somewhat limited, and therefore not covered in this tutorial.

With an external power supply, the motor shield can safely supply up to 12V and 2A per motor channel (or 4A to a single channel).

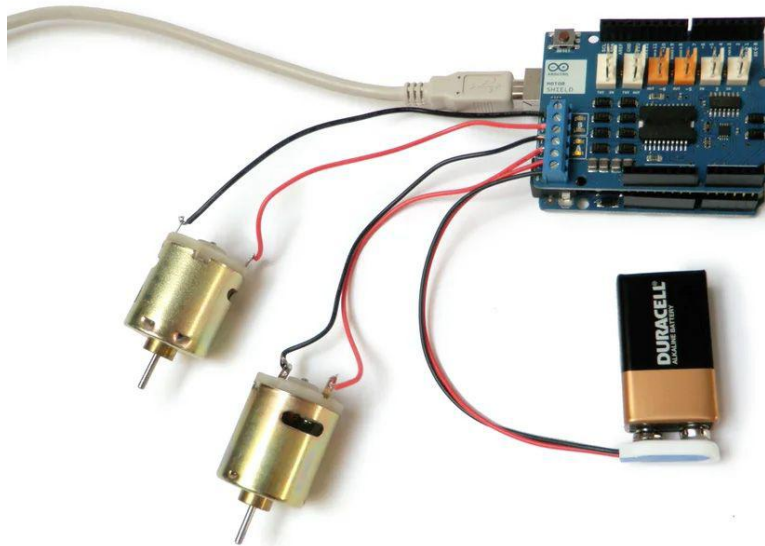
There are pins on the Arduino that are always in use by the shield. By addressing these pins you can select a motor channel to initiate, specify the motor direction (polarity), set motor speed (PWM), stop and start the motor, and monitor the current absorption of each channel.

The pin breakdown is as follows:



Function	Channel A	Channel B
Direction	Digital 12	Digital 13
Speed (PWM)	Digital 3	Digital 11
Brake	Digital 9	Digital 8
Current Sensing	Analog 0	Analog 1

And finally, this is how it should look like when connected:



<http://erasmus-artie.eu>

Find out more in this Instructables page:

<https://www.instructables.com/Arduino-Motor-Shield-Tutorial/>

CONCLUSION

Similar to most living things, our ARTIEBot robot has a brain (microcontroller), eyes (AI camera) and nerves / impulses (motor controller).



Learning scenario 13

Duration: 90 min

Topics

Connecting microcontroller with camera and PC and getting started

Aims

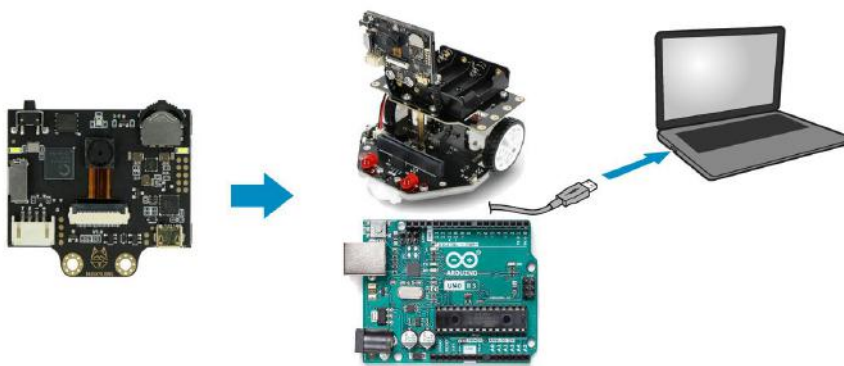
Students connecting microcontroller with camera and PC and observing the operation of the device

Outcomes

Understanding how camera works

Connecting microcontroller with camera and PC and getting started

Before we start programming, we need to connect the camera to the microcontroller (micro:bit or Arduino UNO) and the microcontroller to the computer.



Option 1: Camera > I2Connection > micro:bit/Maqueen plus > USB cable > Laptop or PC

Option 2: Camera > I2Connection > Arduino UNO > USB cable > Laptop or PC

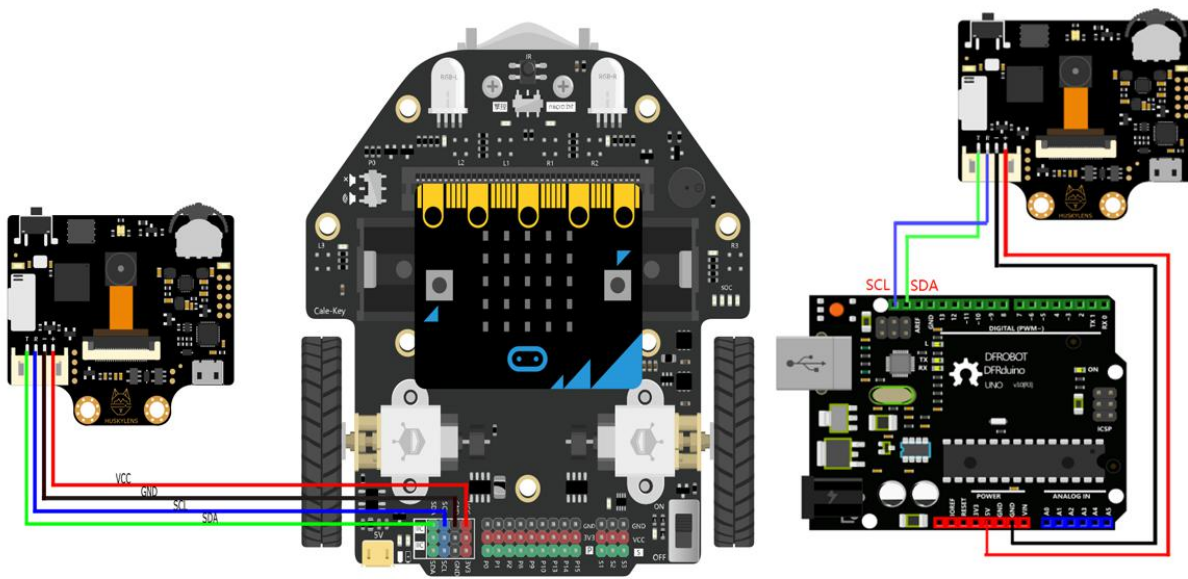


<http://erasmus-artie.eu>

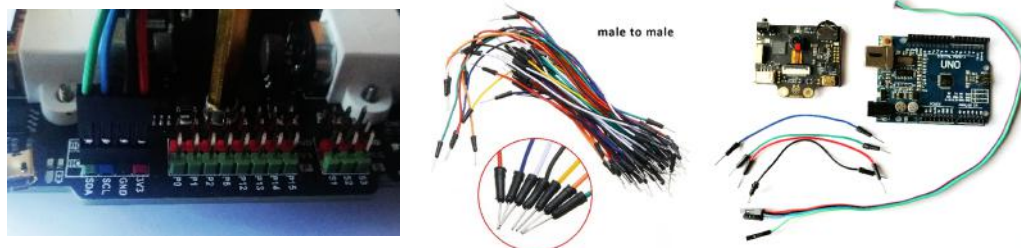


MAIN PART

There isn't much new to say about USB connection, but you probably haven't heard of I2C. The Inter-Integrated Circuit (I2C) bus is a two-wire serial interface originally developed by the Phillips Corporation for use in consumer products. It follows a master/slave hierarchy, wherein the master is defined as the device that clocks the bus, addresses the slaves, and writes or reads data to and from registers in the slaves. The slaves are devices that respond only when interrogated by the master, through their unique address. The I2C bus uses only two bidirectional lines, Serial Data Line (SDA) and a Serial Clock Line (SCL).

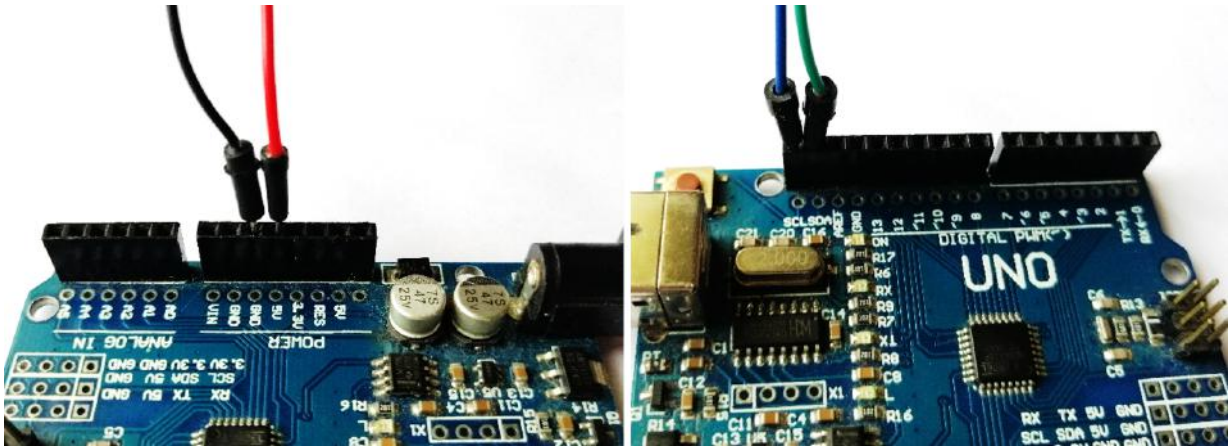


In Option 1 - the solution is very simple and all you have to do is match colours on wires and the connector.

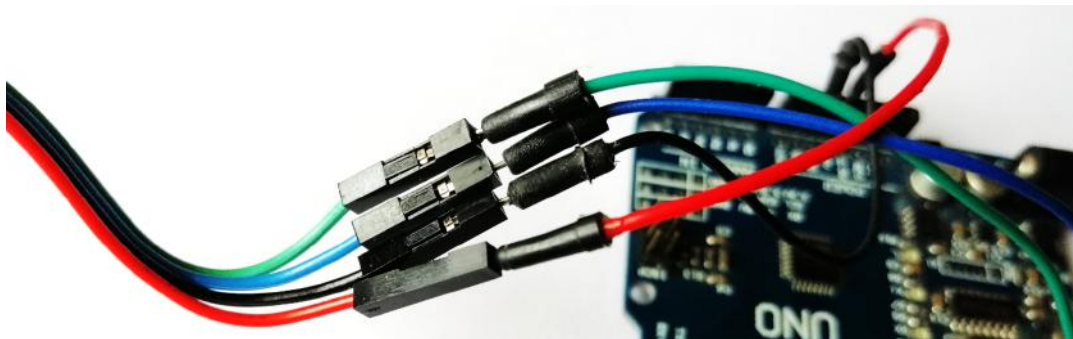


It will be a bit tricky in Option 2 to connect camera cable and Arduino UNO because both connectors are female type so we need 4 jumper wires male to male type which can be found in almost every electronic store which sells DIY parts.



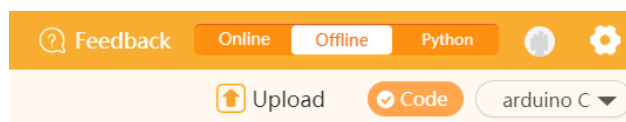


Pick 4 jumpers (red, black, green and blue) to match the corresponding camera pins. First plug the red jumper to 5V pin on Arduino UNO, black jumper to GND, blue jumper to SCL and green jumper to SDA pin as shown in the pictures bellow. The only thing left to do is to plug the other side of these jumpers to camera connector. Match the colours of jumpers to camera connector wires (black to black, red to red, etc...).



You now have your camera connected to the microcontroller. Plug in USB cable to your Maqueen plus robot or Arduino UNO board and connect it with your laptop or PC.

Go to: <http://mindplus.cc/download-en.html> and download version for your computer operating system. Install and start the Mind+. First, switch to **Offline** mode.



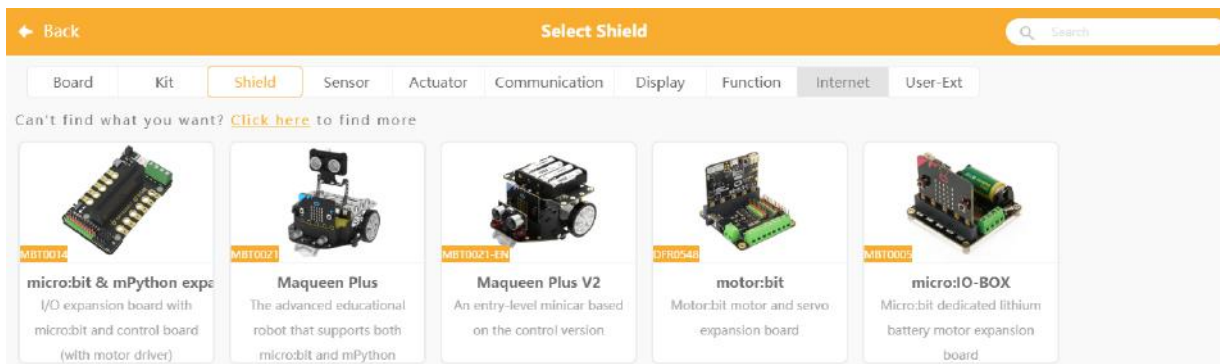


Open Extensions and select Board: Option 1) micro:bit (if you work with) Maqueen plus
Option 2) Arduino UNO



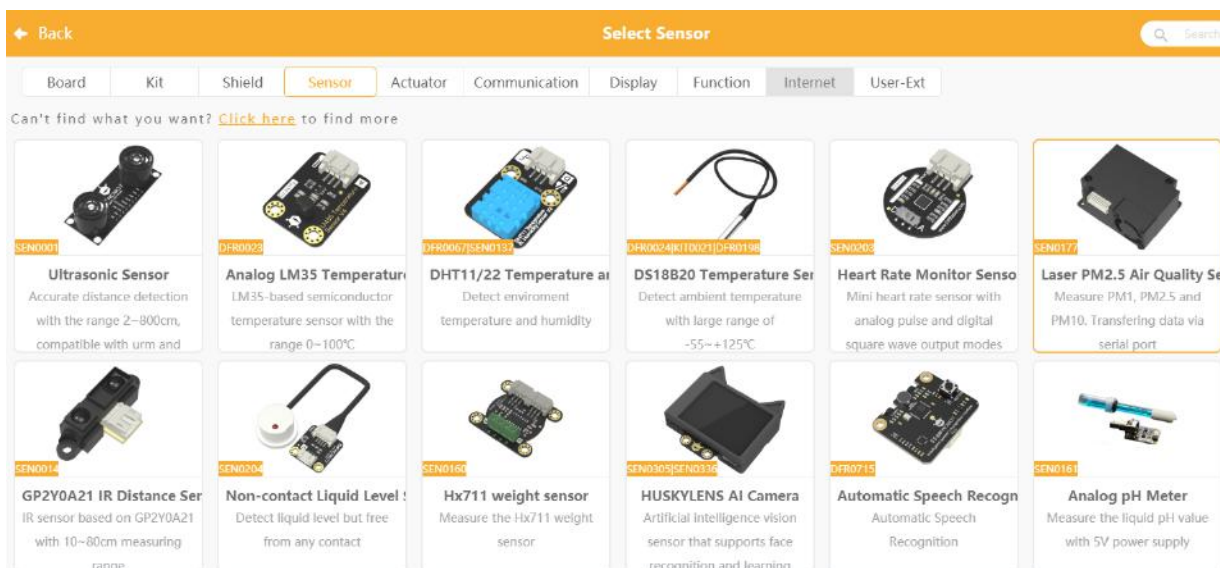
Only for Option 1)

Switch to **Shield** tab and select Maqueen Plus or Maqueen Plus V2 (depending on your version)



Both Option 1) and Option 2)

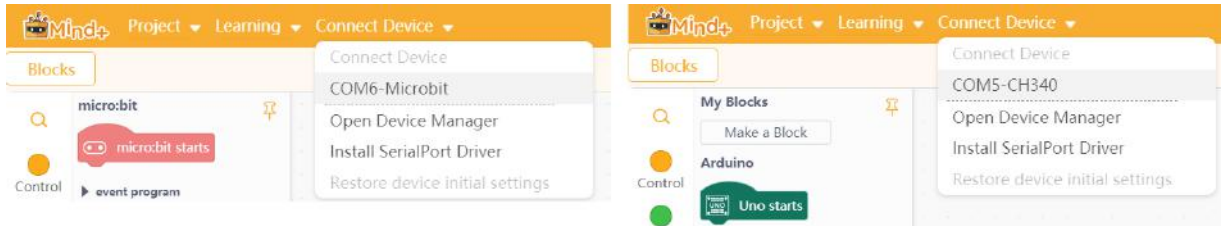
Switch to **Sensor** tab and select sensor - HUSKYLENS AI Camera





After selecting click on <- **Back** and you are ready to use selected board/robot and camera. Let's do a test to see if this works.

Before that you need to connect device. Click on Connect Device and select your port and device

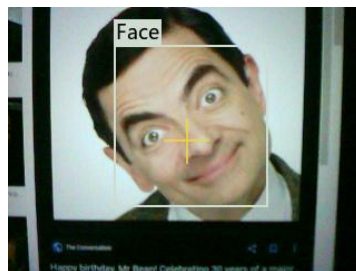


Take the camera and dial the function button to the left until the word "**Face recognition**" is displayed at the top of the screen.



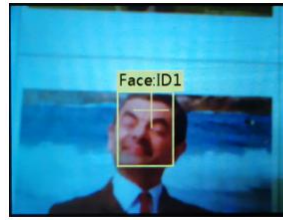
Learning and Detection

1. Face Detection: Point the HuskyLens at any faces. When a face is detected, it will be automatically selected by a white frame with words "Face" on the screen.



Tips: If you want HuskyLens to learn or recognize your face, that is, take a selfie, you can't see the screen at this time, you can determine the status according to the different colours of the RGB indicator.

2. Face Learning: Point the "+" symbol at a face, short press the "learning button" to learn the face. If the same face is detected by HuskyLens, a blue frame with words "**Face: ID0**" will be displayed on the screen, which indicates that HuskyLens has learned the face and can recognize it now.



Visit: https://wiki.dfrobot.com/HUSKYLENS_V1.0_SKU_SEN0305_SEN0336#target_15 for complete Face recognition reference.

Go to Mind+ and start programming. After micro:bit/Uno starts use HuskyLens initialize pin until success block and configure it for I2C as shown on picture below for each option.

```

micro:bit starts
  HuskyLens initialize pin until success
    Communication Method: I2C
    Address: 0x32
    SDA(Green): P20
    SCL(Blue): P19
  
```

```

Uno starts
  HuskyLens initialize pin until success
    Communication Method: I2C
    Address: 0x32
    SDA(Green): A4
    SCL(Blue): A5
  
```

Next, use the block **HuskyLens switch algorithm to Face recognition** for both options.

```

micro:bit starts
  HuskyLens initialize pin until success
  HuskyLens switch algorithm to Face recognition
  
```

```

Uno starts
  HuskyLens initialize pin until success
  HuskyLens switch algorithm to Face recognition
  
```

It is followed by the forever loop which detects if face on camera is recognized as Face ID0. If face is detected in Option 1), you will see face on micro:bit display. If not, you will see X sign. If face is detected in Option 2), it will turn on the onboard LED (D13). If not, LED is turned off.

```

micro:bit starts
  HuskyLens initialize pin until success
  HuskyLens switch algorithm to Face recognition
  forever
    HuskyLens request data once and save into the result
    if HuskyLens check if ID 0 frame is on screen from the result? then
      display pattern
    else
      display pattern
  
```

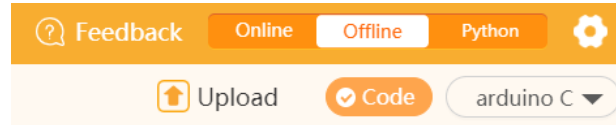
```

Uno starts
  HuskyLens initialize pin until success
  HuskyLens switch algorithm to Face recognition
  forever
    HuskyLens request data once and save into the result
    if HuskyLens check if ID 0 frame is on screen from the result? then
      digital pin 13 output HIGH
    else
      digital pin 13 output LOW
  
```





Press the **Upload** to transfer this code to the micro:bit or Arduino UNO.



After code is transferred, point the camera at the “learned” face and LED on D13 should be turned on. If you move it away from face, D13 is turned off.

If it works as described – everything is fine and ready for use in our mobile AI robot.

CONCLUSION

HuskyLens is an easy-to-use AI computer vision sensor with 7 built-in functions: face recognition, object tracking, object recognition, line tracking, color recognition, tag recognition, and object classification. Through the UART/I2C port, HuskyLens can be connected to Arduino and micro:bit to help you make very creative projects without dealing with complex algorithms.





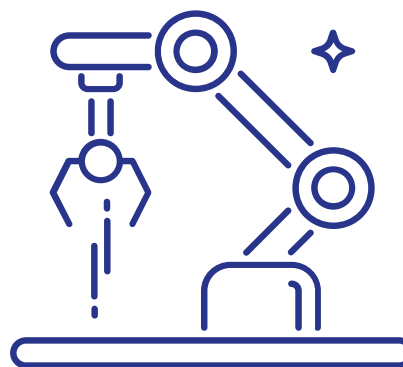
Learning scenario 14

Programming robots - moving

Let's face it, robots are cool. In this class we provide a step-by-step, easy-to-follow tutorial (with code samples) that walks you through the process of programming a basic autonomous mobile robot to move.

Announcement of the goal of the lesson:

To program our ARTIEbot for the first time and to see how to move it.



Duration: 90 min

Topics

Programming robot to move

Aims

To learn how to program a robot to move

Outcomes

Knowing how to write a program for robot to move



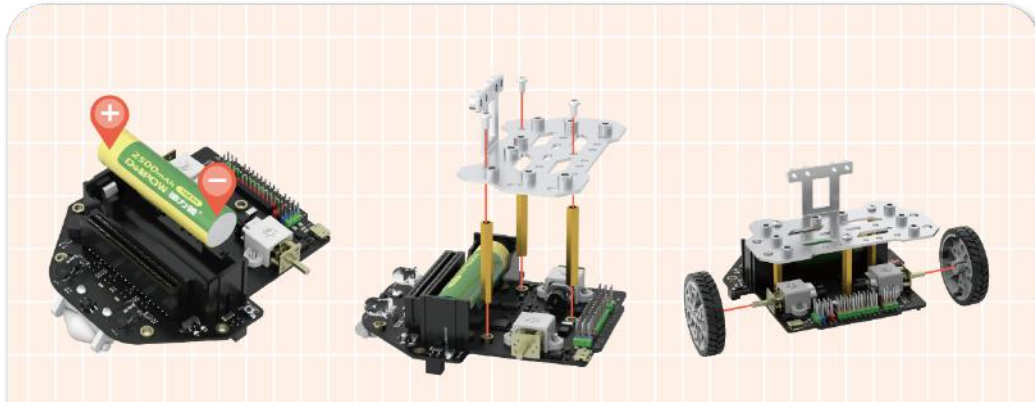
<http://erasmus-artie.eu>

100



MAIN PART

Option 1a - Assembling the V1 Maqueen Plus Robot (recommended for beginners)

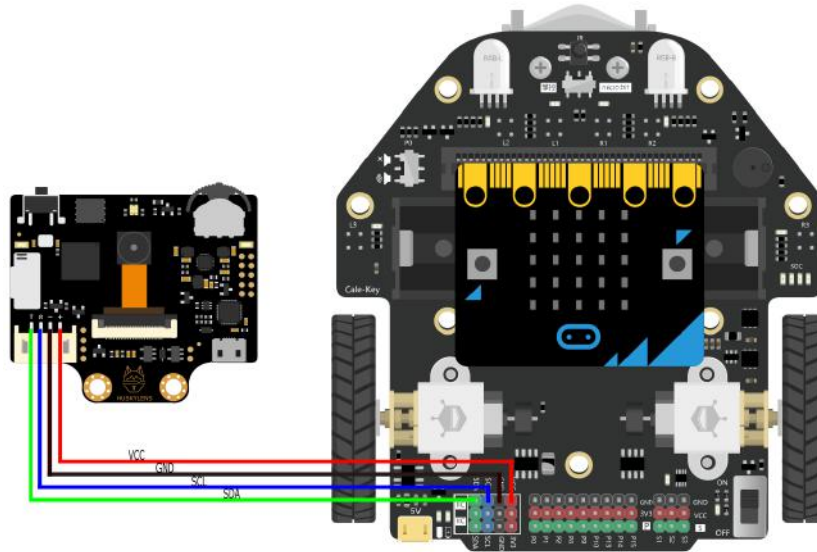


1) insert the 18650 battery 2) install the bracket 3) attach the wheels



4) Mount the HuskyLens camera 5) Place the micro:bit board in the slot





6) Connect the Maqueen Plus interface to the HuskyLens camera

To charge the 18650 battery, use the USB micro port (charging port) on the back of the robot:



Battery indicator

Charging port

When the battery is fully charged, all LEDs light up. The LEDs will turn off one by one as the battery capacity gradually decreases. If all lights go out, the battery needs to be charged. Use a smartphone charger and connect it to the charging port.

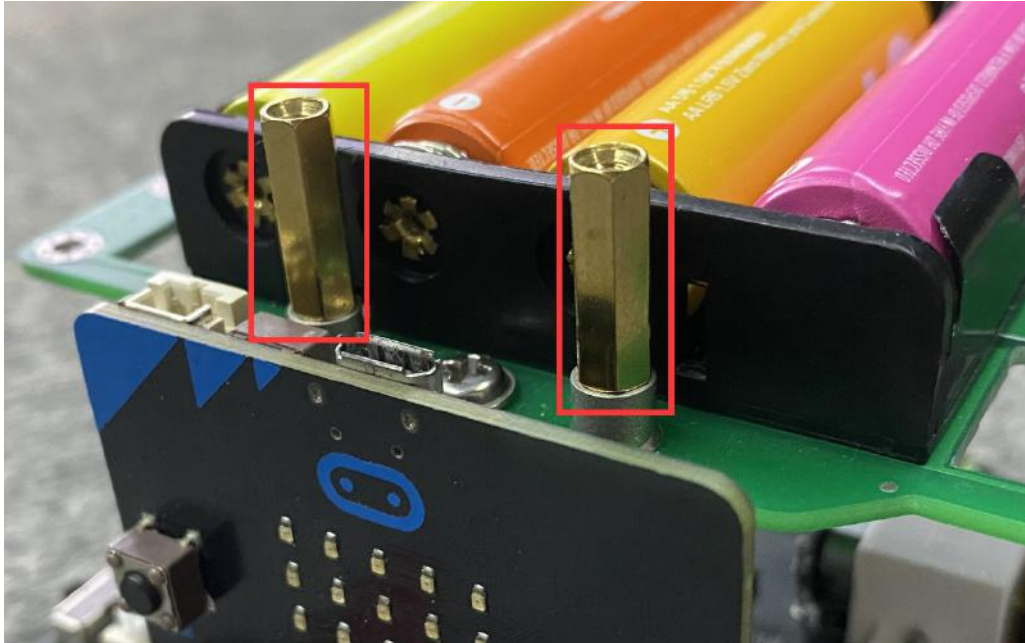
Please note that the charging port is for charging only. To program the Maqueen Plus robot - use the USB micro connector on the micro:bit.

Option 1b - Assembling Maqueen Plus Robot V2 (also recommended for beginners)

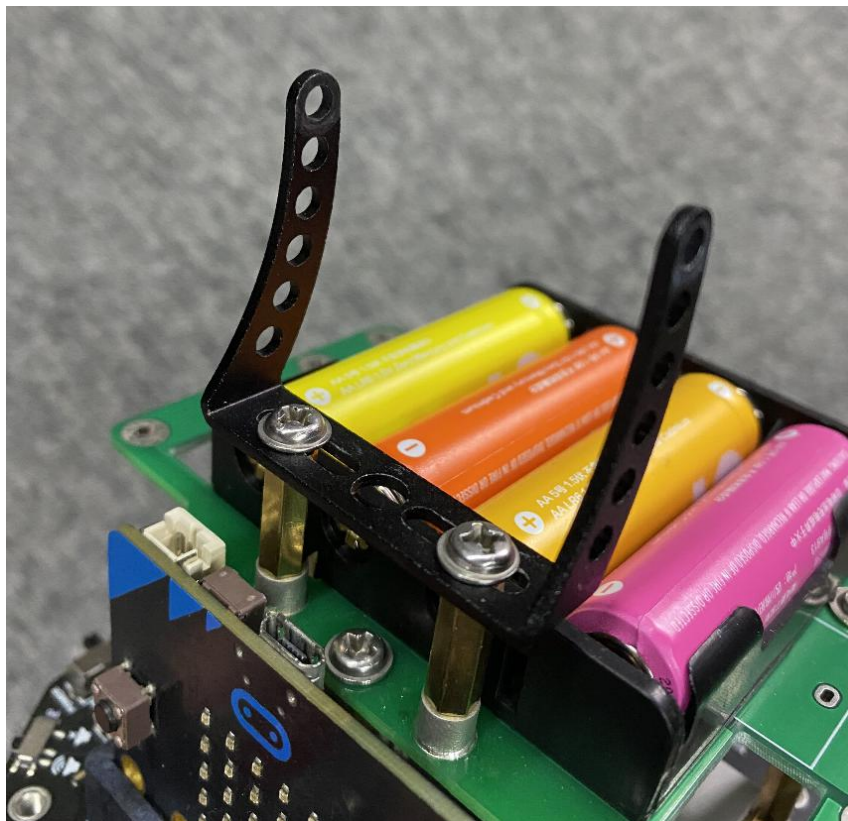
The Maqueen plus V2 robot comes pre-assembled, and you simply need to place the HuskyLens camera on it

1. Install two copper spacers (distances) that are in the kit of materials in the places as shown in the picture.



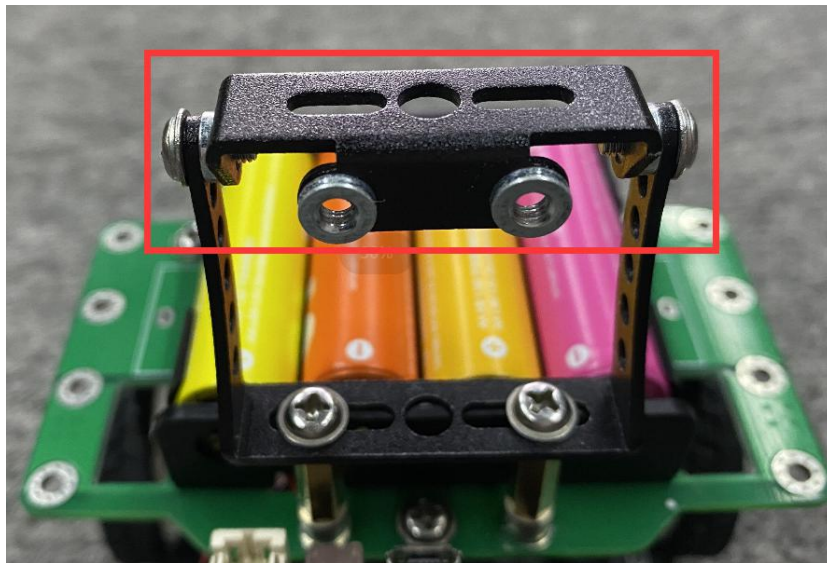


2. Attach the arch bracket (bracket and mounting screws supplied with HuskyLens) to the copper post with screws.

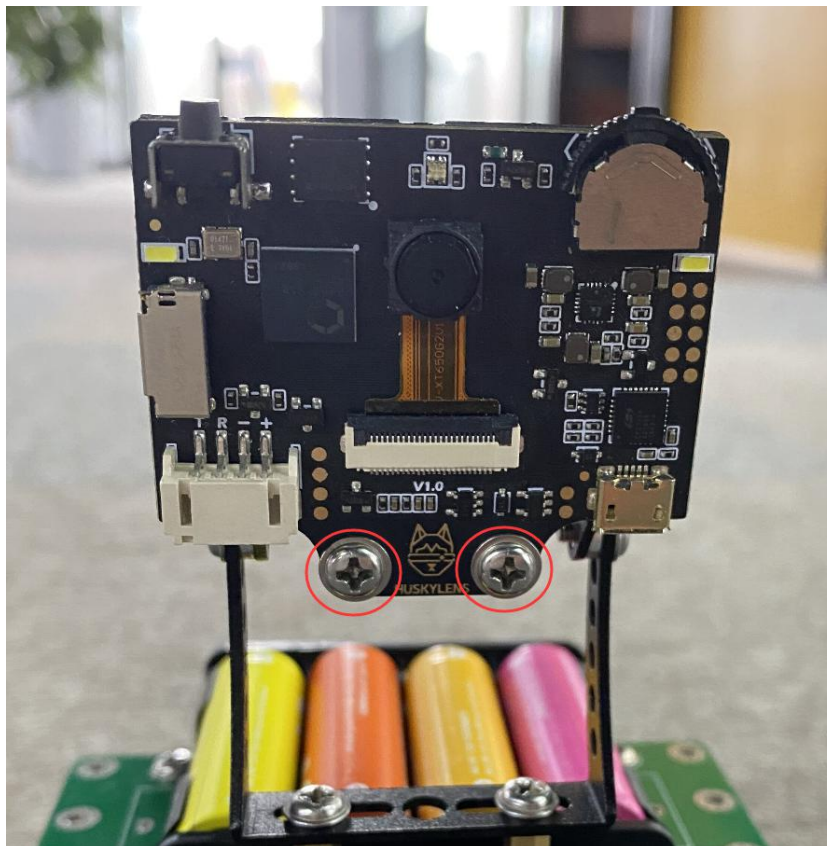




3. Install the second bracket (the bracket and mounting screws are included in the kit supplied with the HuskyLens)

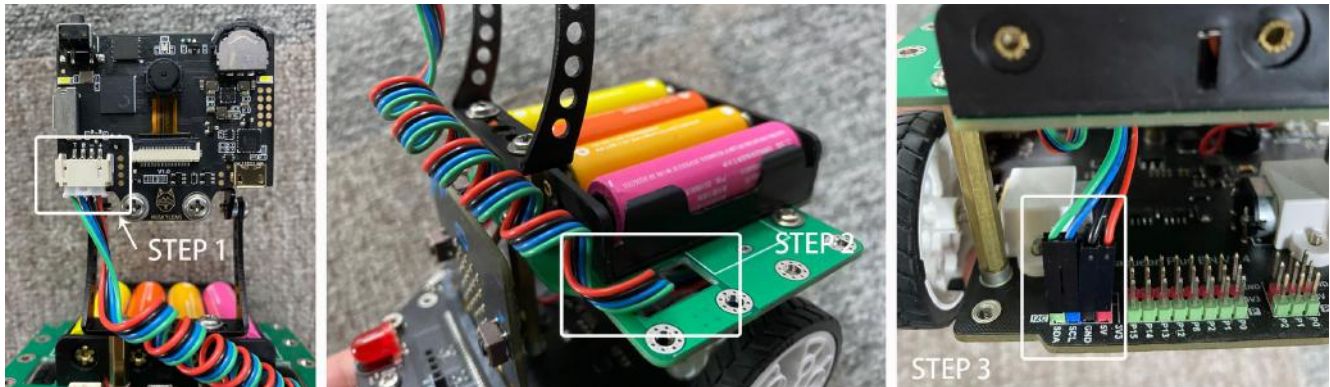


4. Attach the HuskyLens AI camera

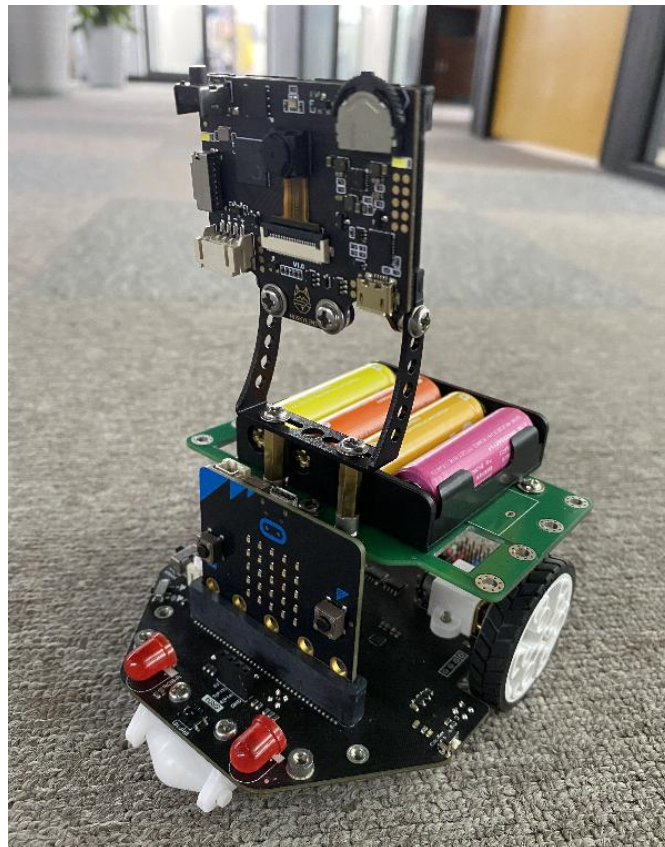




5. Connecting the AI camera to the robot interface



6. Camera installation is complete.





Option 2 - Assembling an Arduino Robot (only for experienced DIY users)

List of materials (component list)

- 1 x Set of materials for a smart car
<https://botland.store/chassis-for-robots/7283-chassis-rectangle-2wd-2-wheel-robot-chassis-with-dc-motor-drive-5904422335649.html>
- 1 x Set of jumpers
<https://botland.store/various-wires/1022-connecting-cables-male-male-65pcs.html>
- 1 x Battery holder (4 x AA)
<https://botland.store/battery-holders/173-battery-holder-4-x-aa-r6-5904422329389.html>
- Various screws and nuts
<https://botland.store/screws-and-nuts/637-screws-nuts-and-washers-set-330pcs-5410329304478.html>
- 1 with Arduino UNO (original or clone)
<https://store.arduino.cc/collections/boards/products/arduino-uno-rev3>
<https://botland.store/arduino-compatible-boards-dfrobot/2683-dfduino-uno-v3-compatible-with-arduino.html>
- 1 x Arduino motor shield R3 (Arduino motor shield R3)
<https://store.arduino.cc/collections/shields/products/arduino-motor-shield-rev3>
- 1 x HuskyLens AI camera
<https://store.arduino.cc/collections/dfrobot/products/gravity-huskylens-ai-machine-vision-sensor>
- 8 x AA batteries (try using rechargeables)
- Cable ties (if you want to fix wires and battery holders)

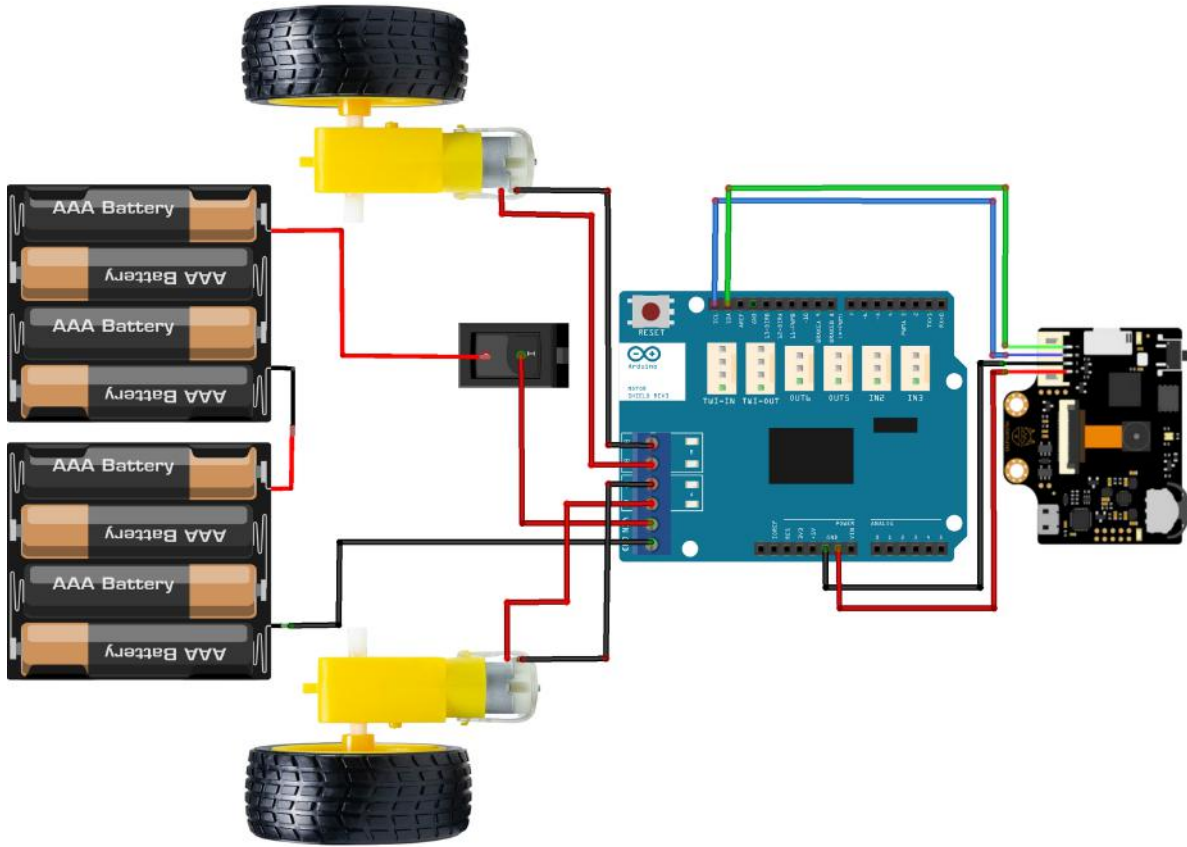
Necessary tools

- Screwdrivers (you will need one flat and one Phillips)
- Lemilica
- Insulation tape or heatshrink tubes
- Cutting pliers

Scheme

Here is the wiring schematic. Notice that we will be using a dual battery holder in a series connection so that each of the batteries in the holder is connected (+ to - and - to +). 8 AA batteries should provide more voltage to power the consumer and prevent the Arduino or HuskyLens from resetting when the motors start running. Alternatively, you can use 3 18650 batteries in a series connection.





Smart Car Material Kit - Open the plastic bag of parts

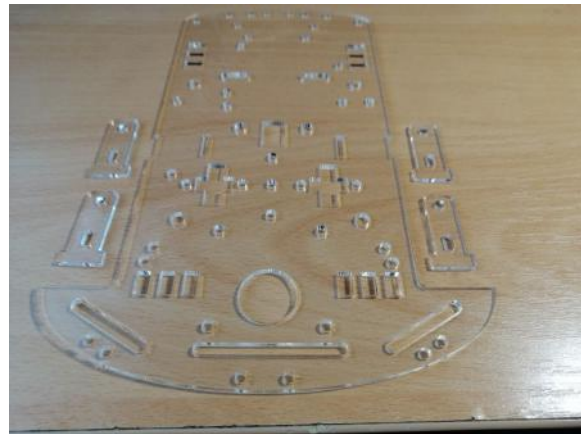


<http://erasmus-artie.eu>

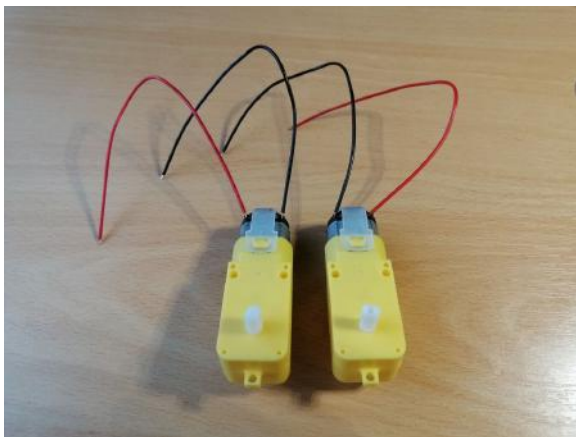




Peel off the protective film from the plate and motor mount



Solder the wires (or jumpers) to the motors



Watch this short video to see how it's done:

<https://www.youtube.com/watch?v=xSWKnnvGWBs>

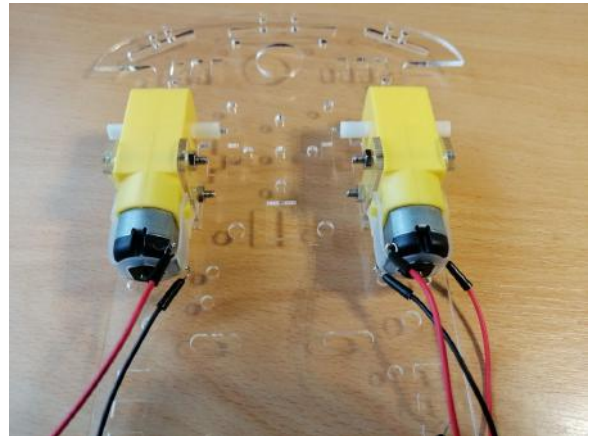
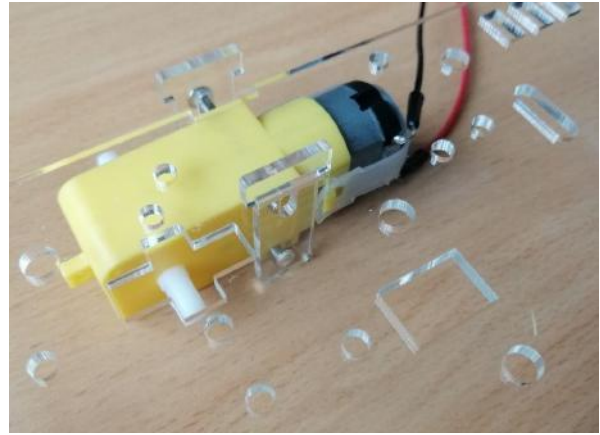
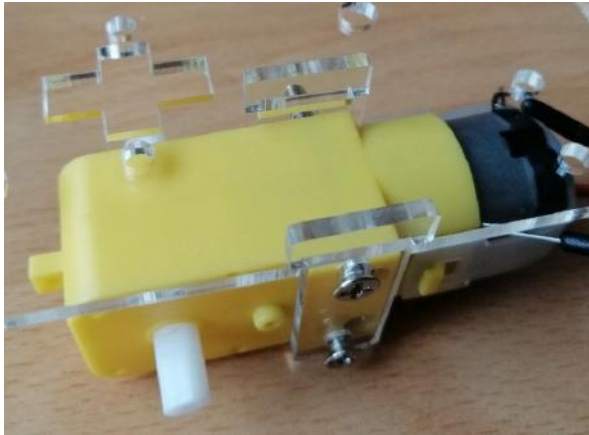
Ask your teacher or parent to help you with this if you don't have previous experience or tools.



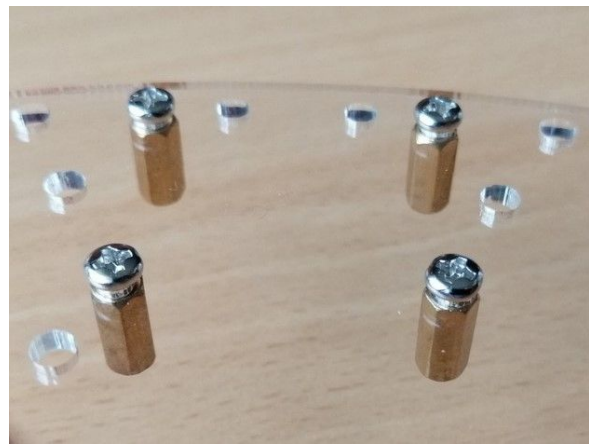
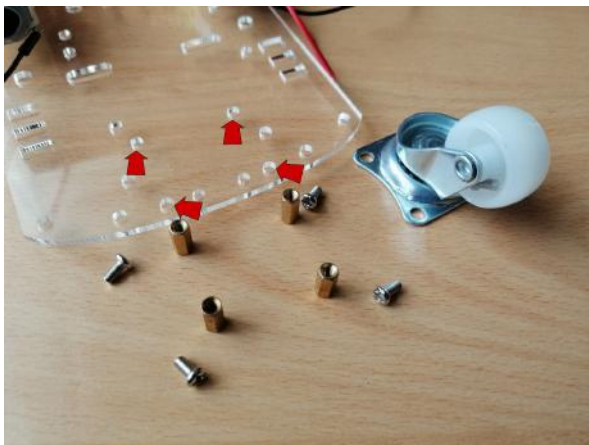
<http://erasmus-artie.eu>

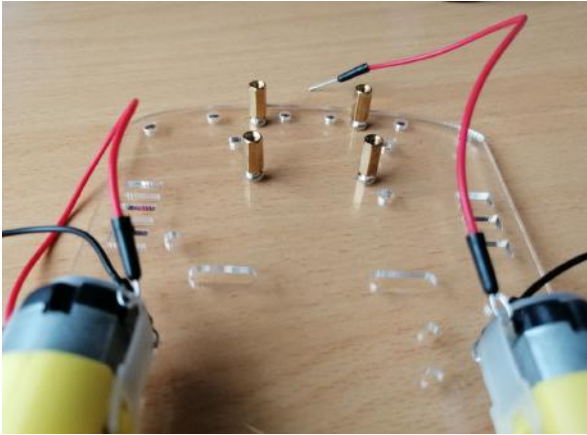


Attach the motors to the "T brackets" using bolts and nuts

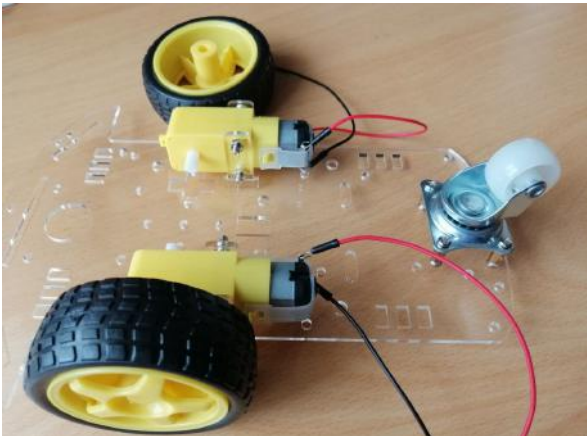


Fasten the small (rear) wheel using screws and spacers.

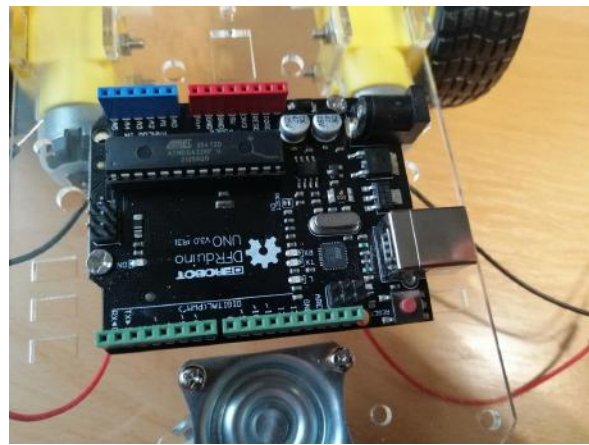
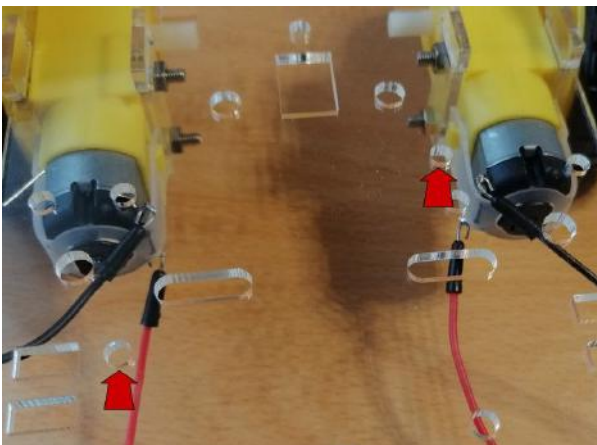


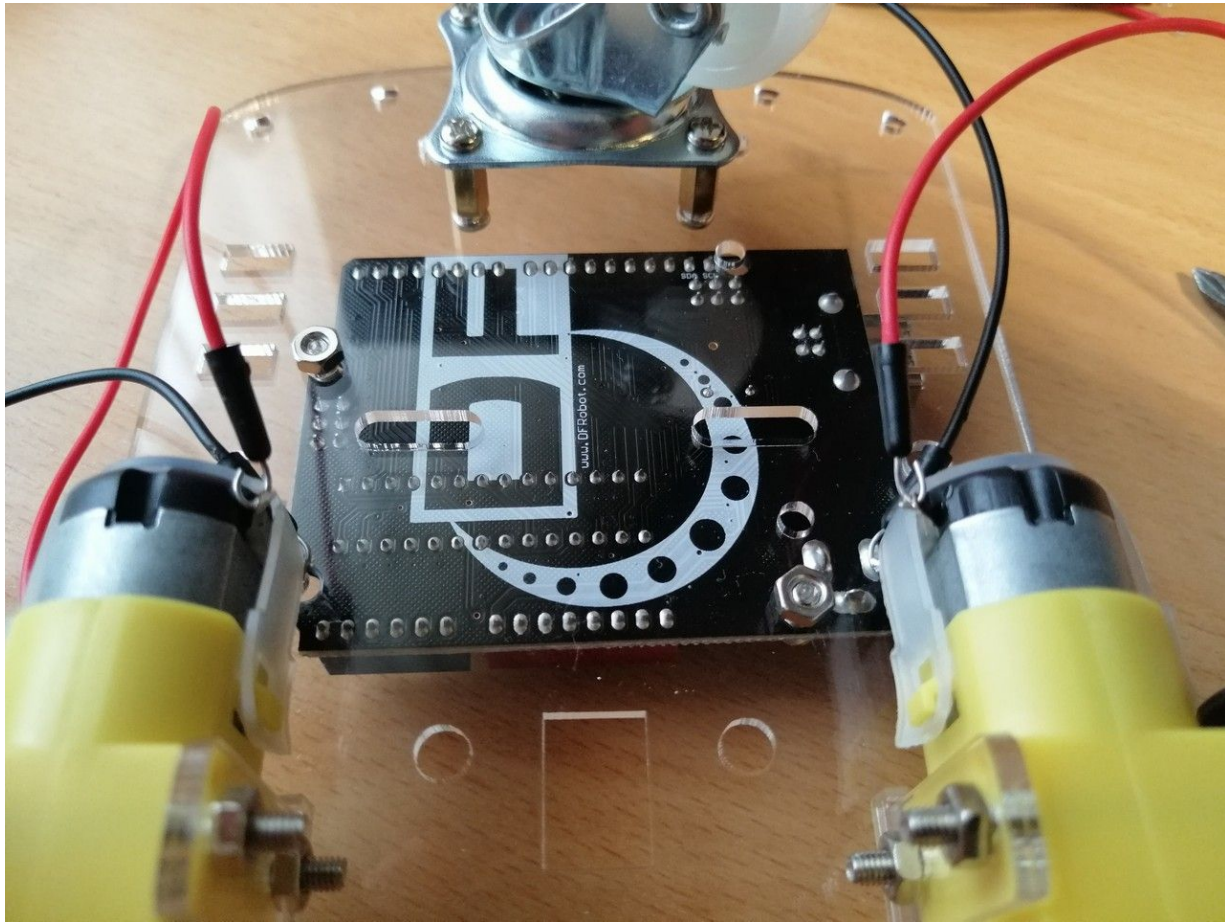


Attach the wheels to the motor axles - look at the shape of the axle and the shape of the wheel opening



Attach the Arduino UNO board to the marked holes



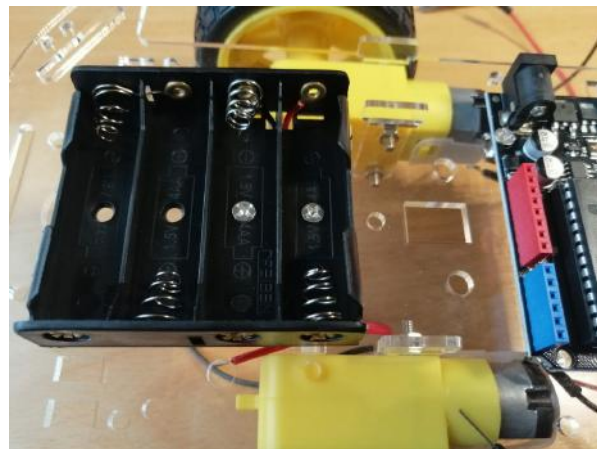
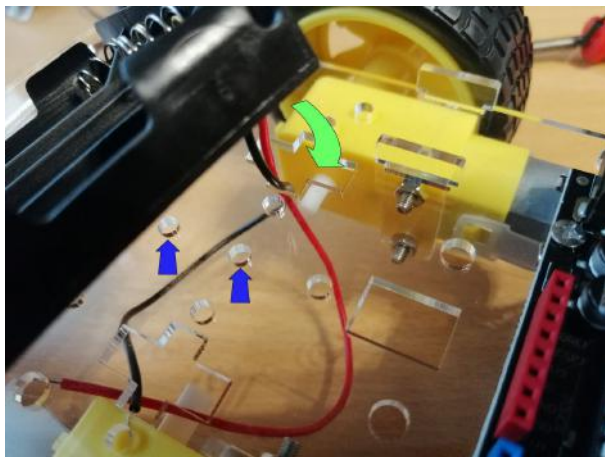


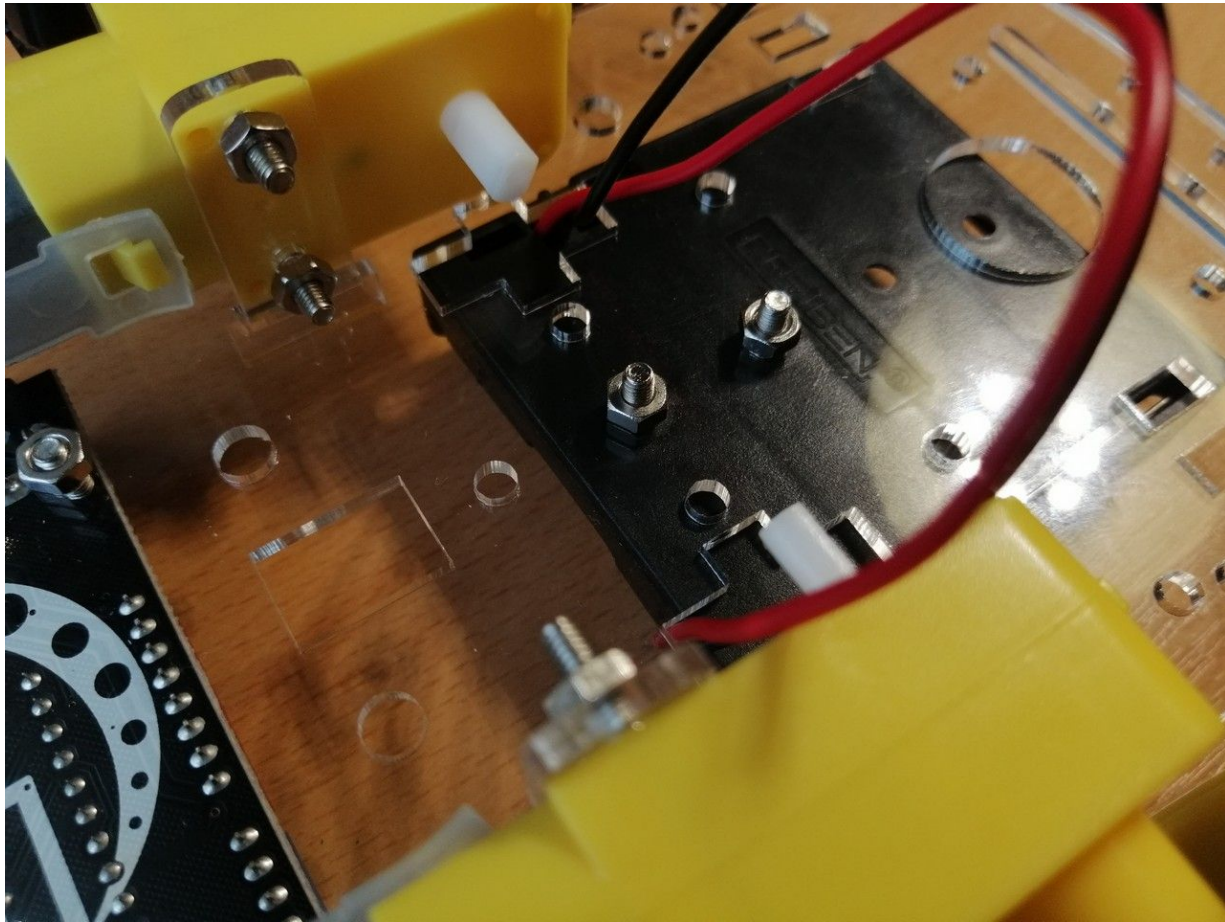
You can see some details in this video, although we take a slightly different approach:

<https://www.youtube.com/watch?v=3a-bE1VlaU8>

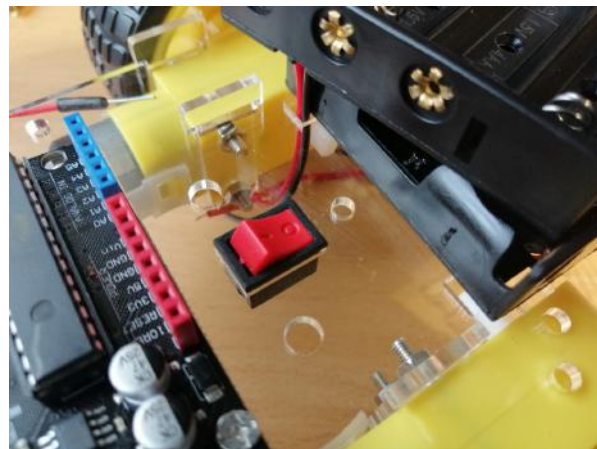
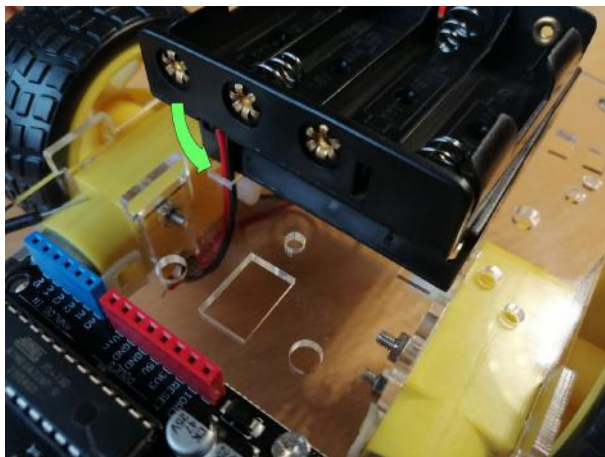
Step 10) Attach the first (bottom) battery holder using the screws in the marked holes (blue arrows)

First, thread the bracket cables through the hole (marked with a green arrow)



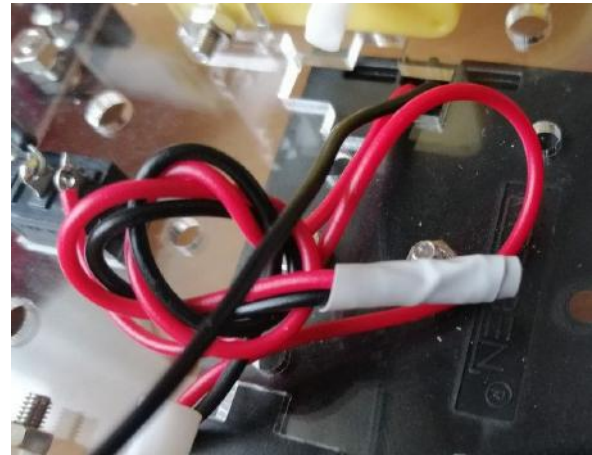
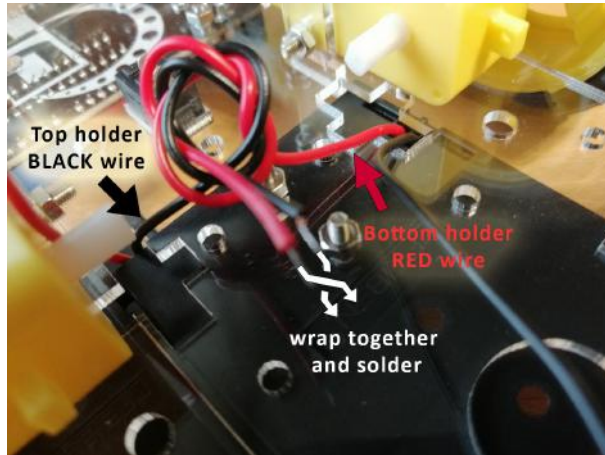


Pull the cable of the second (upper) battery holder through the marked hole (green arrow)
Place the switch in the rectangular hole located between the Arduino UNO and the battery holder

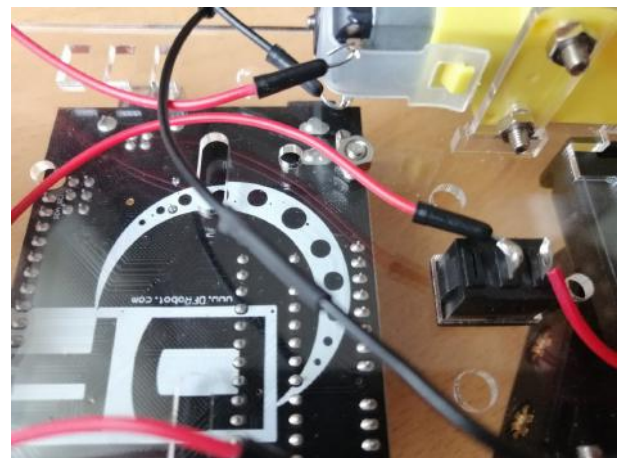
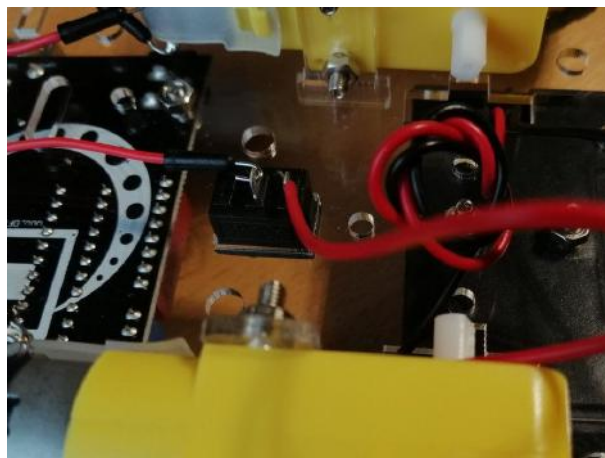




Connect the red wire from the lower battery holder to the black wire from the upper battery holder
Insulate the joint with heat-shrink tubing or insulating tape.

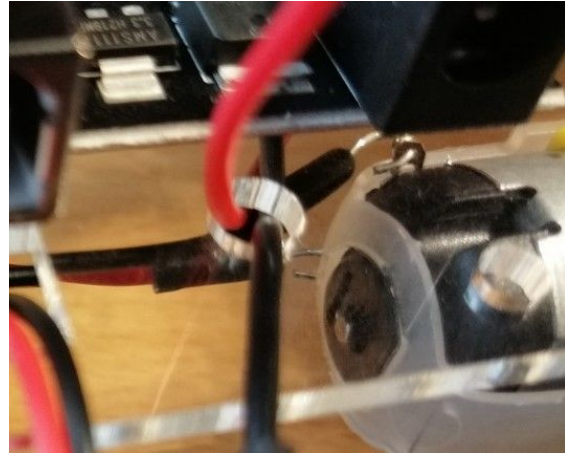
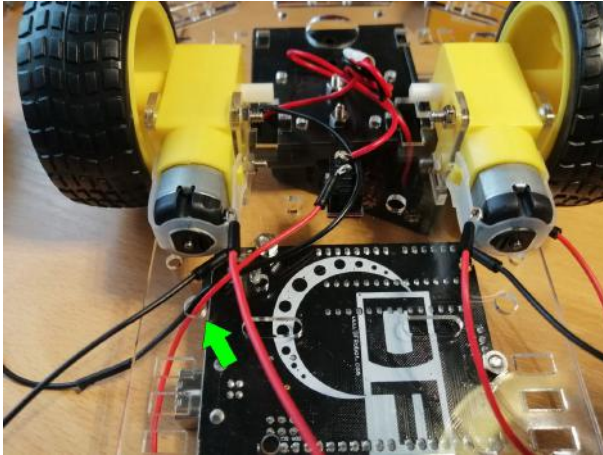


Solder the red wire from the bottom bracket to the switch
After that, solder the red jumper (or wire) to the second contact of the switch. Solder the black jumper to the bottom bracket of the black wire. Insulate the joint with heat-shrink tubing or insulating tape.

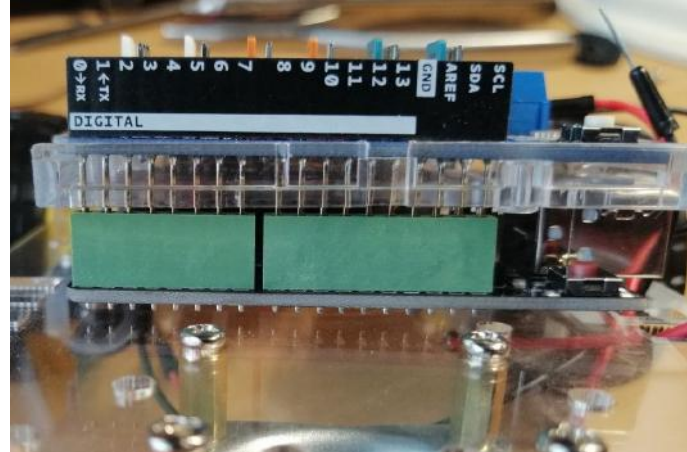
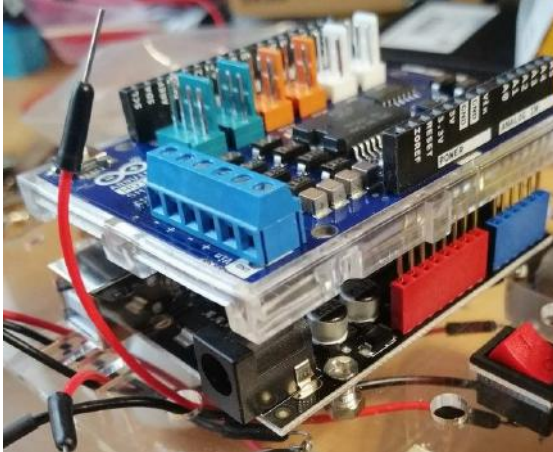




Pull the black and red power wires through the marked hole (green arrow)



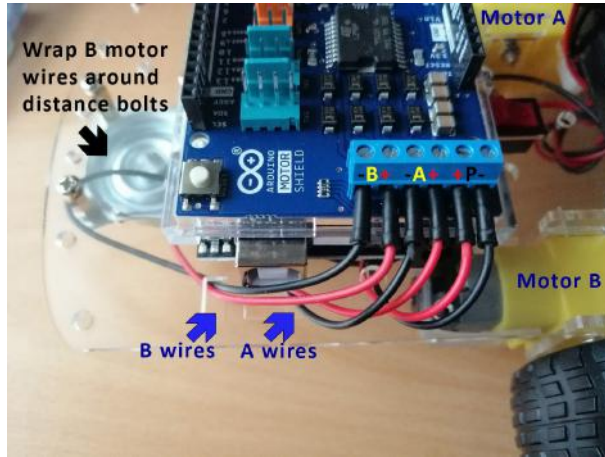
Place the Arduino motor shield into the Arduino UNO board
Align the legs (pins) of the shield and the Arduino UNO and push the shield down.





Thread the motor wires through the marked holes.

Shorten the ends of the jumpers with cutters and connect them to the shield terminals for the motors by tightening the screws as indicated (- Motor B + - Motor A + + Power P -).



Open the HuskyLens AI camera box and take out the kit of materials

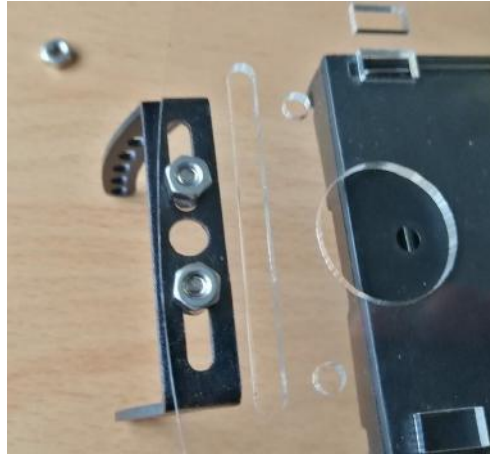


Attach the smaller bracket to the camera - use the screws from the kit





Attach the larger camera mount to the robot body



Connect the two brackets together with screws and connect the camera cable

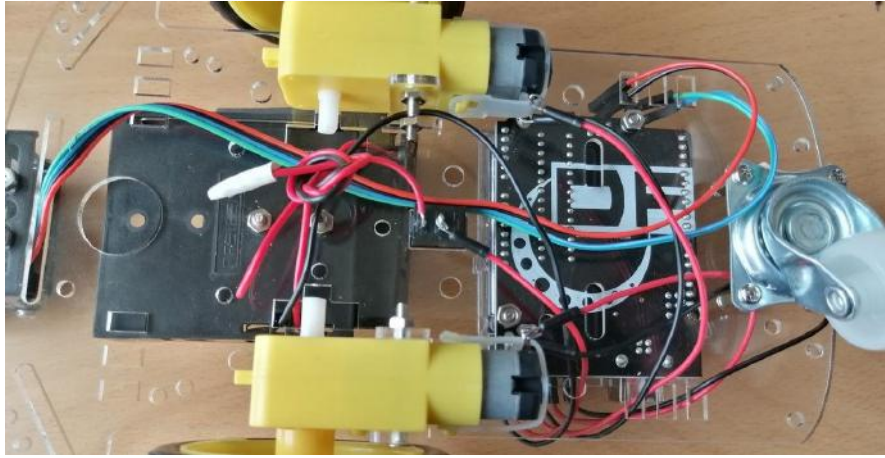


Pull the camera cable through the hole

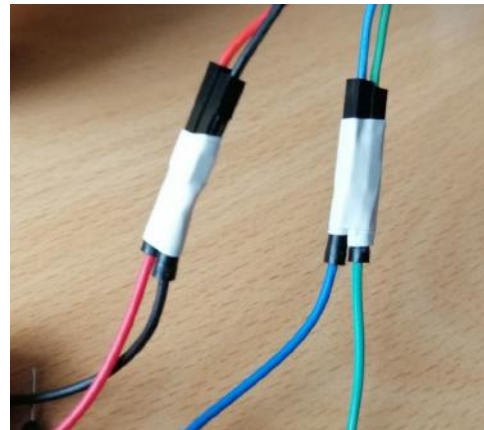
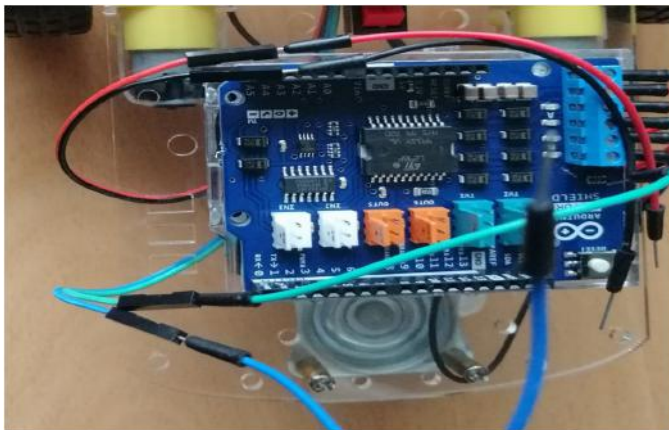




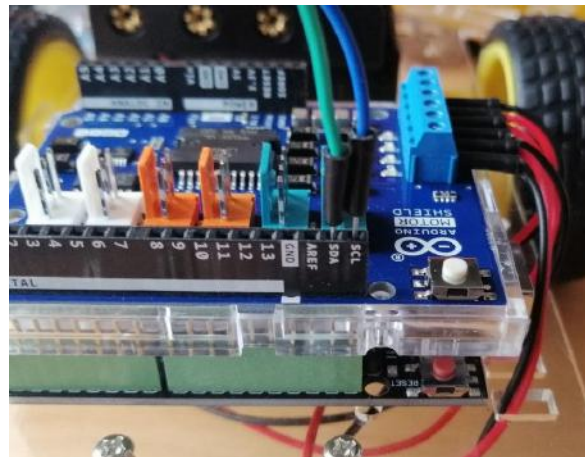
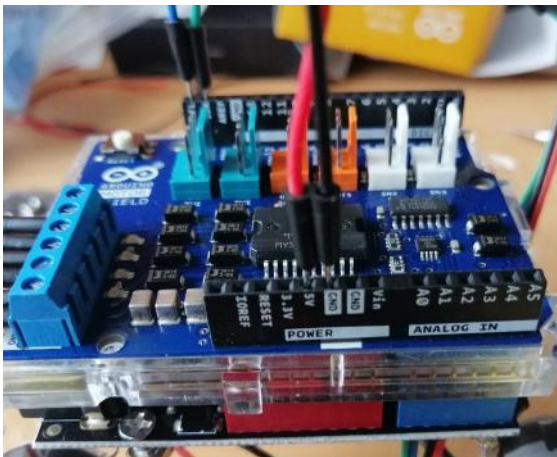
Turn the robot over and thread the camera cable through the holes as shown in the image below



Connect the jumpers to the camera cable - use jumpers with matching colors
Secure the joint with insulating tape



Attach the power jumpers and I2C connections to the motor shield
Red jumper goes to 5V, black to GND, blue to SCL, and green to SDA port

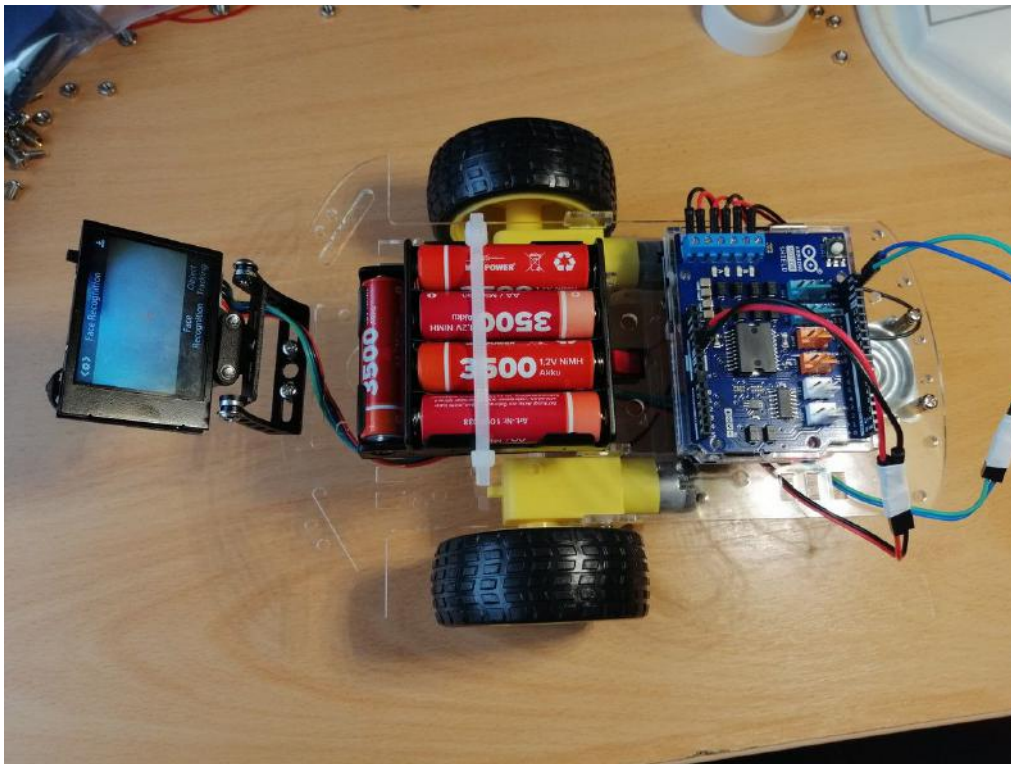




Place the batteries in both holders
Use 2 cable ties to secure the top and bottom battery holders (don't over tighten - you should be able to move the top holder back and forth)



Turn on the switch - if you connected everything correctly - the Arduino UNO and the HuskyLens camera will turn on.





Learning scenario 15

Duration: 90 min

Topics

Programming robot for object following using object tracking and object recognition

Aims

To learn how to program a robot for object following

Outcomes

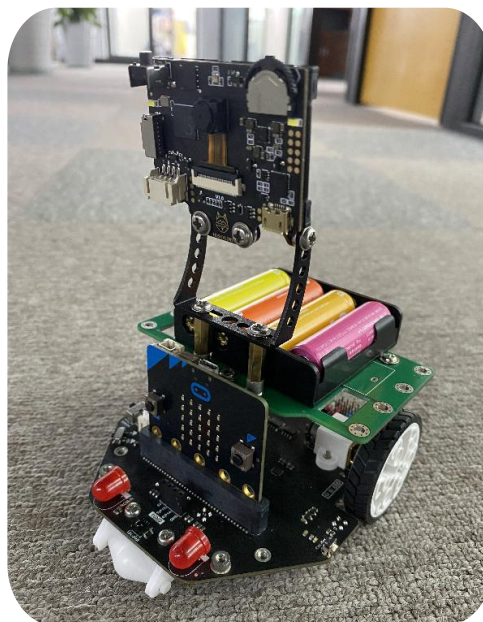
Knowing how to write a program for a robot to be able to follow an object

Programming a robot - object following

Robots are cool. In this lesson, we show step-by-step, easy-to-understand examples of programming the movements of an autonomous mobile robot.

Announcement of the goal of the lesson:

Write our first program with which we will start the robot.

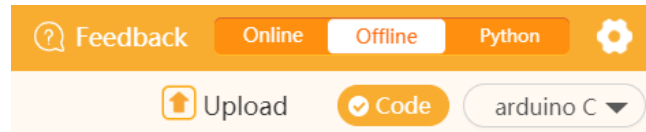




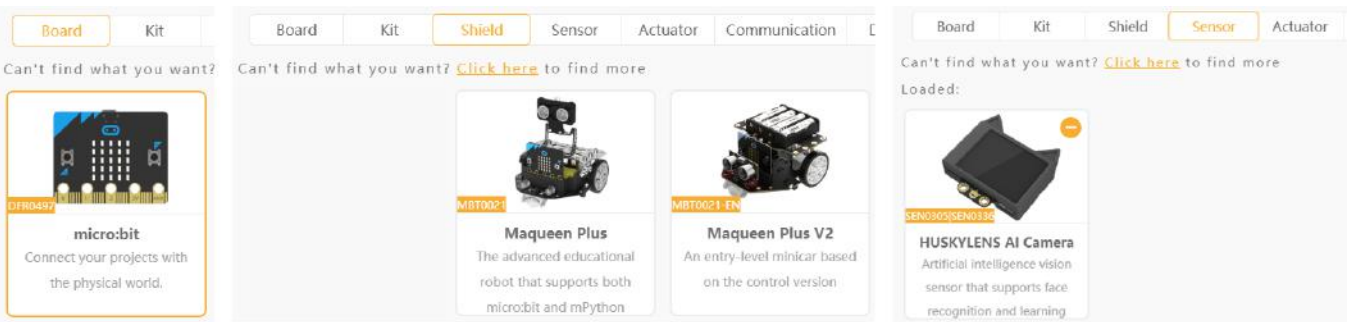
MAIN PART

The teacher assists the students in writing the code for the movement of the robot we assembled.

Step 1 (Both Options): Go to: <http://mindplus.cc/download-en.html> and download the version for your computer operating system. Install and run Mind+. After starting, switch to offline mode (Offline).



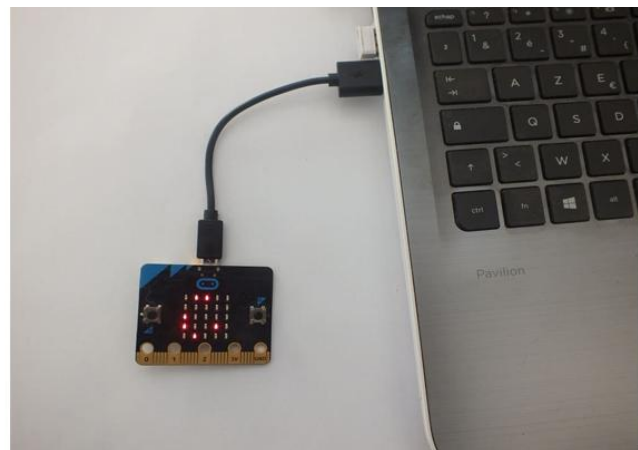
If you are working with an Arduino, skip directly to step 2. Maqueen Plus



Click on **Extensions** and on the **Board** tab select **micro:bit**, on the **Shield** tab select Maqueen Plus or Maqueen Plus V2 and on the **Sensor** tab select HUSKYLENS AI camera. Click on **Back** and your program is ready to use the selected modules. Connect the micro:bit to your computer via a micro USB cable and the power indicator on the board will turn on

<http://erasmus-artie.eu>

120





Click on Connect Device and select micro:bit. If necessary, install device drivers. Test the transfer to the micro:bit with these blocks:

```

micro:bit starts
set all motor direction rotate forward speed 200
wait 1 seconds

```

Click on **Upload**. The robot should move forward for a second and stop. Try this sequence - the robot should move as described in the comments. (forward 2 seconds, backward 2 seconds, counterclockwise rotation 2 seconds and clockwise rotation 2 seconds after which the robot stops)

```

micro:bit starts
set all motor direction rotate forward speed 200
wait 2 seconds
set all motor direction rotate backward speed 200
wait 2 seconds
set left motor direction rotate backward speed 200
set right motor direction rotate forward speed 200
wait 2 seconds
set left motor direction rotate forward speed 200
set right motor direction rotate backward speed 200
wait 2 seconds
set all motor stop

```

Drive forward for 2 seconds

Drive backward for 2 seconds

Turn counter-clockwise for 2 seconds

Turn clockwise for 2 seconds

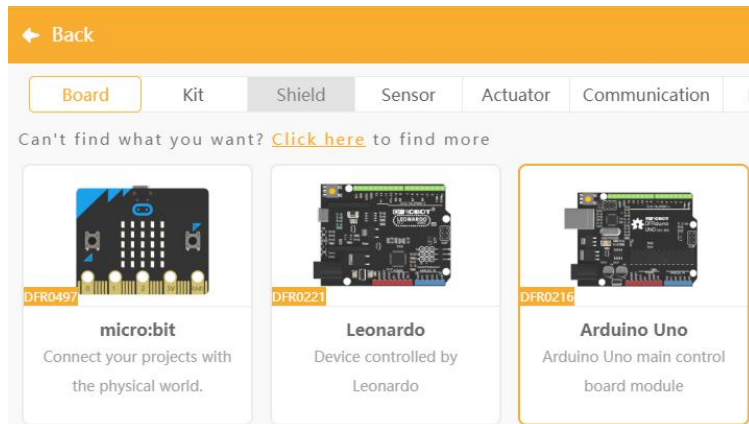


<http://erasmus-artie.eu>

Click on **Upload**. The robot should move as described in the comments (yellow blocks). Try different speeds to make your robot go faster or slower.

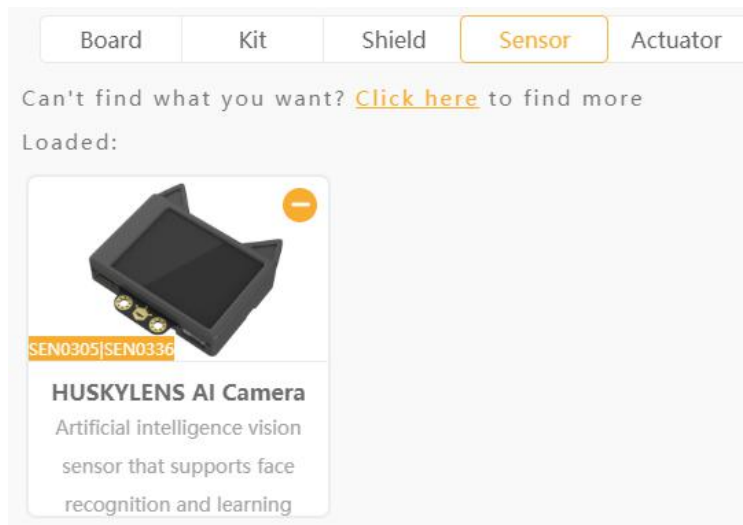


Step 2: Open Extensions and select **Board** - Arduino UNO

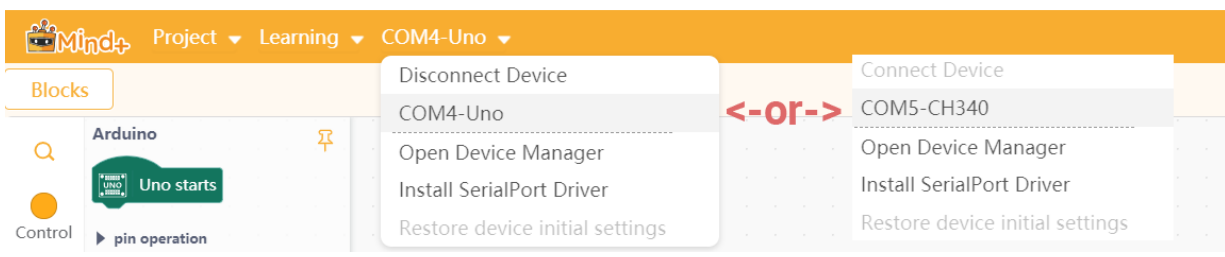


S

Step 3: Go to the Sensor tab and select the sensor - HUSKYLENS AI Camera



Step 4: Once selected click on <- **Back** and you are ready to use Arduino and Sensor blocks. Let's test it to see if it works. Before that, you need to connect the device. Plug in your Arduino UNO via USB cable and select COM X-Uno (or CH340) depending on the Arduino manufacturer.



http://erasmus-artie.eu



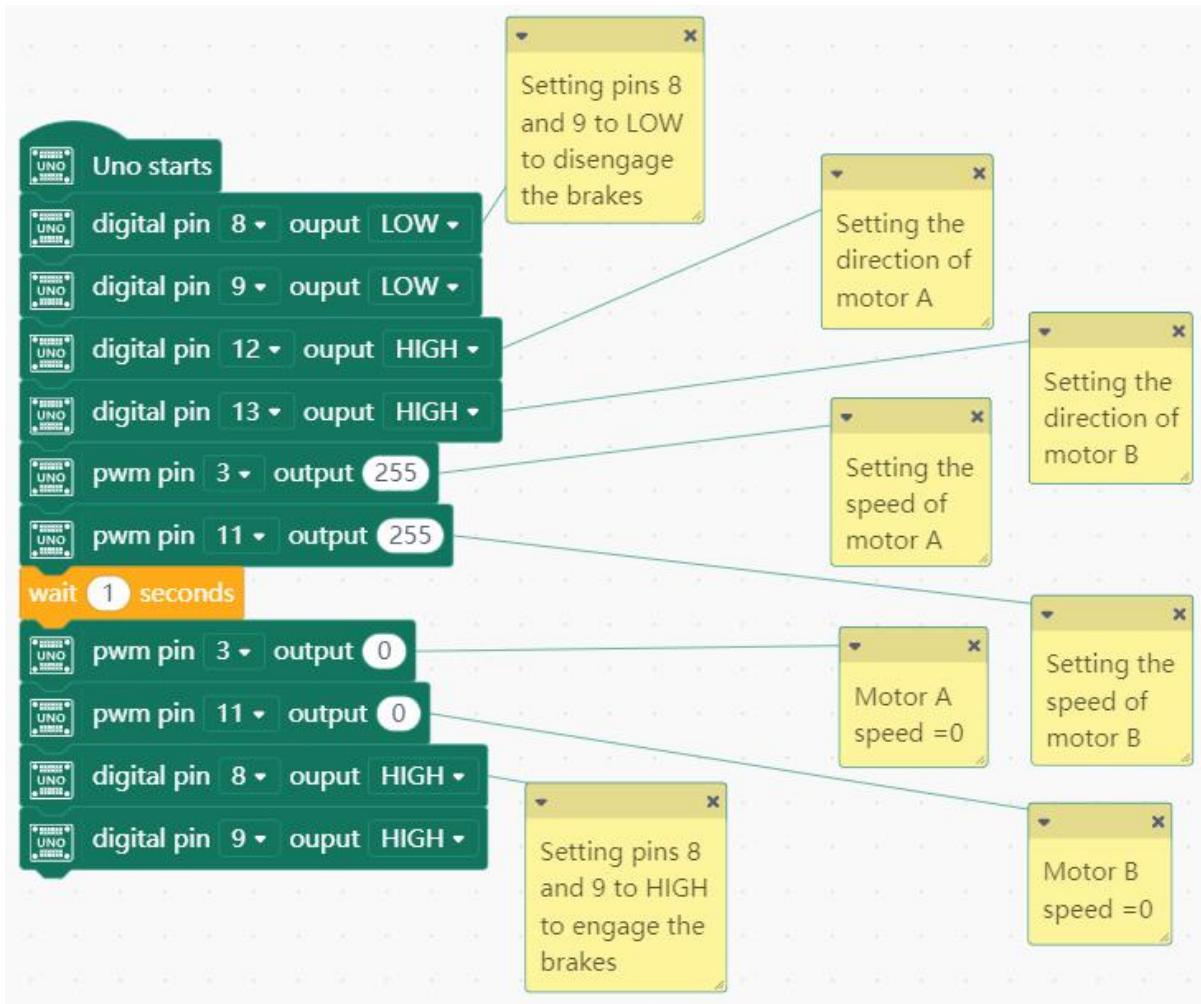
Basic movements

Remember this table from the hardware deployment scenario?

Function	Channel A	Channel B
Direction	Digital 12	Digital 13
Speed (PWM)	Digital 3	Digital 11
Brake	Digital 9	Digital 8
Current Sensing	Analog 0	Analog 1

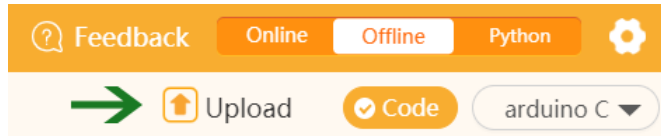
The two engines (A and B) are left and right engines. Digital pins 12 and 13 are used to change the direction (values HIGH - one direction and LOW - the opposite direction), and PWM pins 3 and 11 are used to adjust the speed (values 0-255). Pins 9 and 8 enable/disable braking (HIGH - brakes on, LOW - brakes off).

Below you can see a sample code with comments on the right to help you understand how it works.





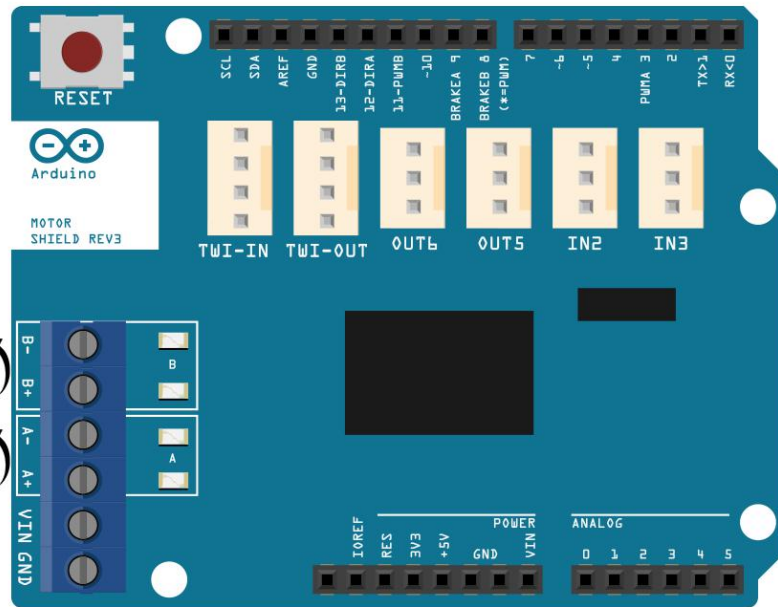
Step 1a: Press Upload to upload this code to the Arduino UNO.



Step 2a: Be careful, the robot will start moving immediately after the transfer is complete! The robot should move forward for a second and stop.

Troubleshooting - if the robot:

- drive in reverse - replace the red and black wires in the connectors for both motors.
- rotates clockwise - swap the red and black wires in the B motor connectors.
- rotates counterclockwise - swap the red and black wires in the A motor connectors.



switch wires for motor B ↻

switch wires for motor A ↻



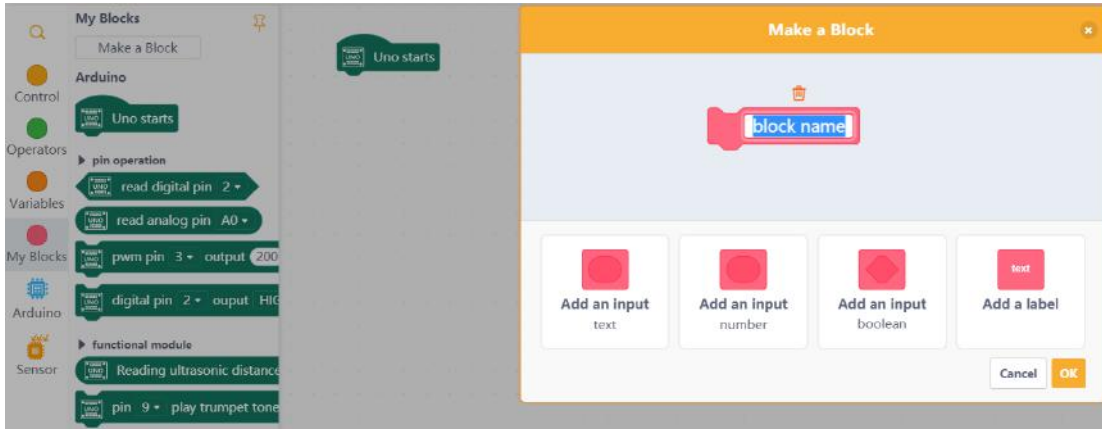
<http://erasmus-artie.eu>

You should now have your robot driving forward with a HIGH state on pins 12 and 13. There's also an easier way to program the motion - we'll use our custom blocks instead of repeating a whole bunch of blocks to control the pins.

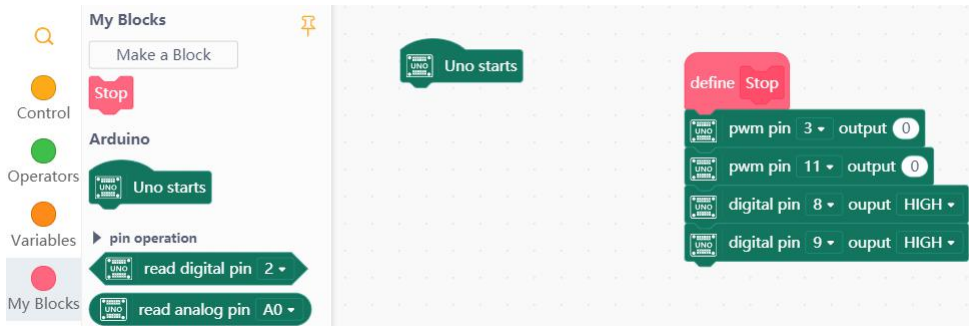
Step 1b: Click on the **My Blocks** group (red).

Step 2b: Click the **Make a Block** button and change "block name" to "Stop".

Step 3b: Click OK.



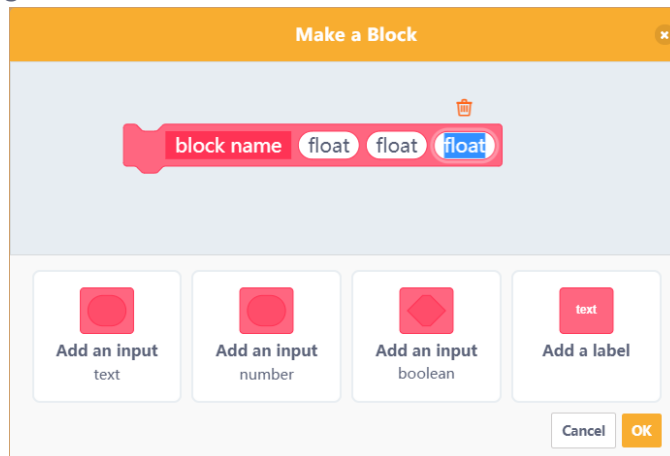
You've created your first block, but it's not connected to anything right now, so we need to define it first.



<http://erasmus-artie.eu>

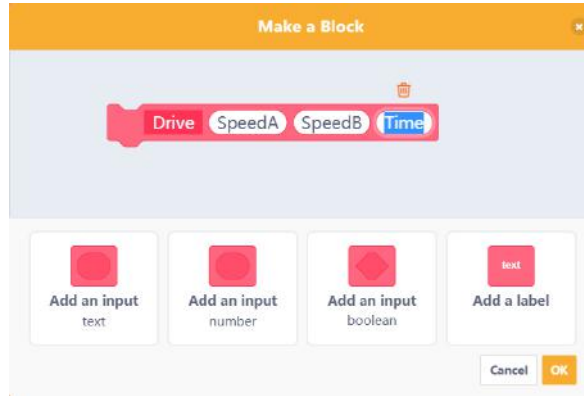
The stop block contains 4 blocks. The first two blocks set the speed of both motors to 0, and the last two are blocks that, with a HIGH state, brake the motors. If you want to use it to coast the motors without coasting, just set the state of digital pins 8 and 9 to low (LOW).

Now it's time to create an input block with 3 numeric parameters that will contain both motor speed and duration. Click the Create Block button again, then click Add an input - number - 3 times and you should get this:





Change the name of the block to *Drive*, change the first float to *SpeedA*, the second float to *SpeedB*, and the third float to *Time*, then click the **OK** button.

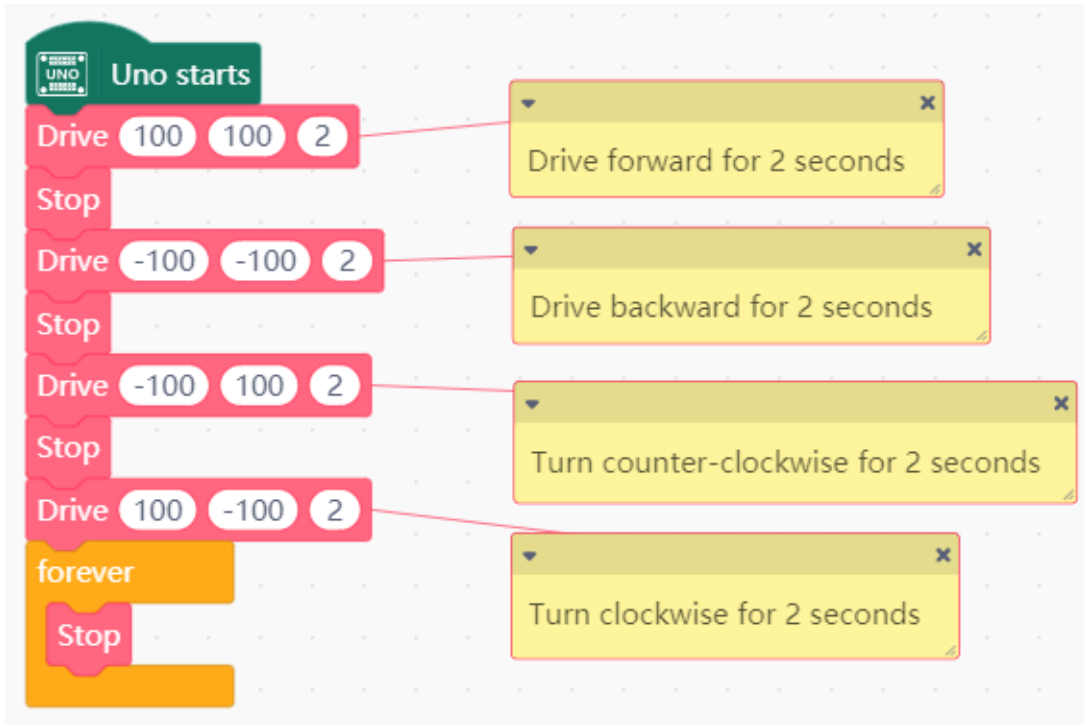


The basic idea is to get the SpeedA and SpeedB values (acceptable range: -255 to 255), then check if either number is negative (or both) and if so - reverse the direction by setting the value for the corresponding pin. You have to use the absolute value on the PWM pin to start the motor.

```
define Drive SpeedA SpeedB Time
  digital pin 8 output LOW
  digital pin 9 output LOW
  if SpeedA > 0 then
    digital pin 12 output HIGH
  else
    digital pin 12 output LOW
  if SpeedB > 0 then
    digital pin 13 output HIGH
  else
    digital pin 13 output LOW
  pwm pin 3 output abs of SpeedA
  pwm pin 11 output abs of SpeedB
  wait Time seconds
```



So let's try to run our robot with the following blocks.



f <http://erasmus-artie.eu>

The last stop block is in a non-jumping loop, which ends the movement of the robot. Change the values in the **Drive** block, then upload your program to the Arduino UNO and watch how fast your robot moves. You are now ready to use your ARTIEbot in more complex examples, including those with the HuskyLens camera.

CONCLUSION

Let's face it, robots are cool. They may rule the world one day, and hopefully at that point they will have mercy on their puny creators (the robotics engineers) and help us build heaven in space. It was a joke, of course, but he knows it.



Learning scenario 16

Duration: 90 min

Topics

Programming robot for object following using object tracking and object recognition

Aims

To learn how to program a robot for object following

Outcomes

Knowing how to write a program for a robot to be able to follow an object

Programming a robot - object following

We learned how to make a robot move in our previous lesson.

Check with your students if they understand everything from the last class and if they are ready for the next step.

Let's move our robots toward a specific object now. First, we must detect that object and track it. Tracking a moving object requires visual object tracking technology as well as manual operation.

Announcement of the goal of the lesson:

How to prepare a robot for tracking objects?



<http://erasmus-artie.eu>



MAIN PART

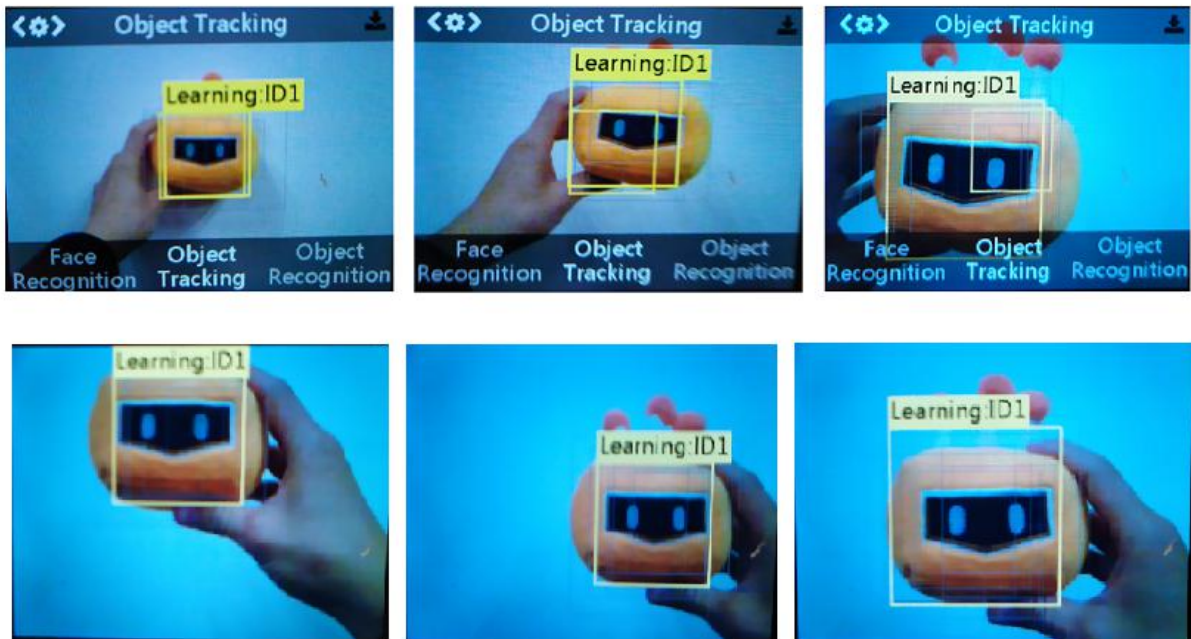
Object tracking is an important assignment in computer vision. It refers to the process of continuously inferring the state of objects in video sequences. The image is collected by a single camera and the image information is transmitted to a microcontroller. After analysing and processing, the relative position of the moving object is calculated. At the same time, the robot carrying camera is controlled to rotate and track the object in real time.

- When the object tracking system performs the tracking function, it is mainly divided into 4 steps:
- object recognition
- object tracking
- object movement analysis
- controlling the robot (or any other system) with camera

Object recognition - Learning

Connect the micro:bit or Arduino UNO with HuskyLens camera to your laptop or desktop computer. Point HuskyLens to the target object, adjusting the distance and until the object is contained in the orange bounding box of the centre of the screen. It is also acceptable that only part of the object is included in the box but within distinct features.

Then long press "learning button" to learn the object from various angles and distances. During the learning process, the orange box with words "Learning: ID1" will be displayed on the screen.





When HuskyLens can track the object at different angles and distances, then release the "learning button" to complete the learning. If there is no orange box on the centre of the screen, it means that the HuskyLens has already learned an object. If you want to track another object - select "**Forget Learned Object**" and learn again.

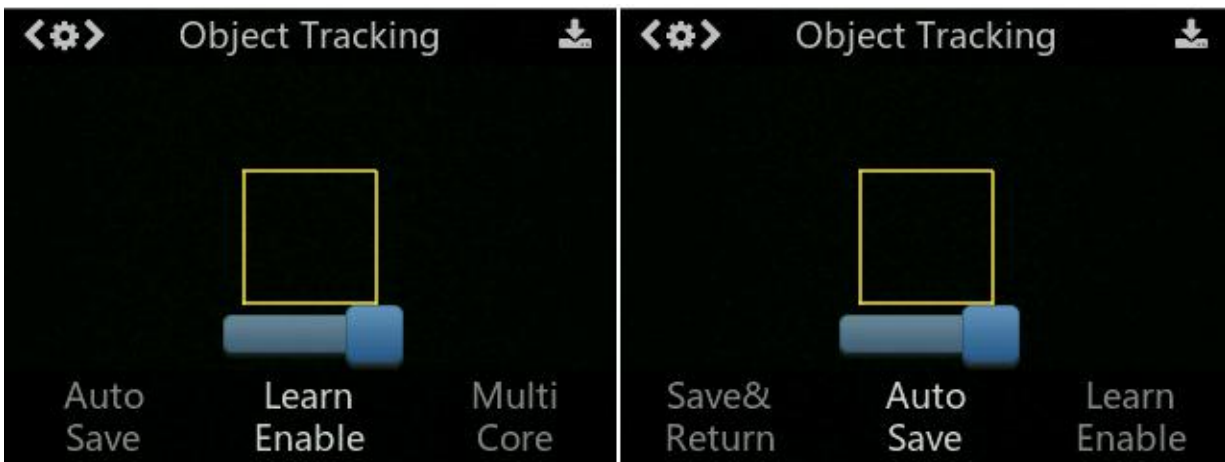
Under the object tracking function, HuskyLens can keep learning, that is, as long as the camera sees the learned object, it will keep learning the current state of the object, which is conducive to capturing dynamic object. Operation method: Long press the function button to enter the parameter setting of the object tracking function.



f

Push the function key to the right to get to the Learn Enable parameter, then press the function key briefly to select it, then push it to the right to turn on **Learn Enable**, that is, the slider on the bar is in the right position. Then briefly press the function key again to confirm this parameter. Exit the submenu with a short press on the **Save&Return** item to save the changes.

<http://erasmus-artie.eu>



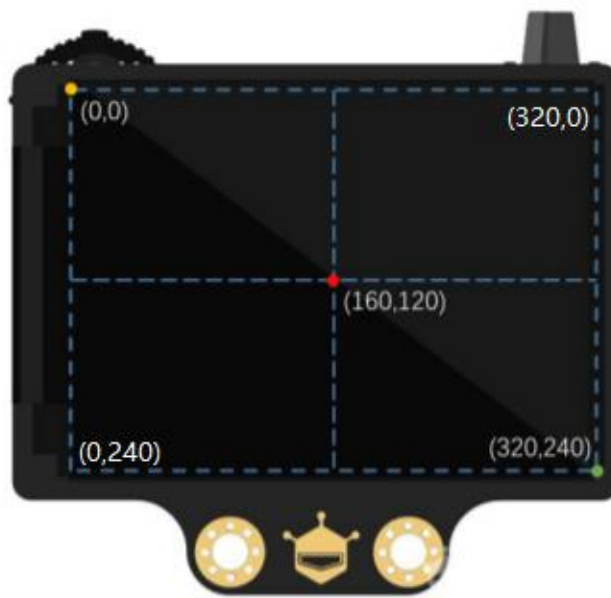


When restarting HuskyLens, the last learned object is not saved by default, and you can turn on the **Auto Save** switch as previously explained.

How to work: Same as above, after entering the parameter setting, move the slider in the Auto Save item to the right position. That way, you only need to learn the object once. Restarting the camera will save the object you last learned.

Object tracking

HuskyLens sensor screen resolution is 320*240, as shown in the following picture.



The coordinates of the object centre point obtained through the program are also within this range. For example, if the coordinate values obtained are (160, 120), the object being tracked is in the centre of the screen.

"X coordinates" and "Y coordinates" refer to the position of the box centre point in the screen coordinate. "Object width" and "Object height" refer to the size of the frame. Under the object tracking function, the frame is square, so the width and height are equal.

Test the object tracking - Option 1 (Maqueen Plus/HuskyLens)

Open your Mind+ and load extensions for work with Maqueen Plus and HuskyLens camera. Use this code:



```

micro:bit starts
HuskyLens initialize pin until success
HuskyLens switch algorithm to Object tracking
set motor All stop

forever
  HuskyLens request data once and save into the result
  if HuskyLens check if ID 1 is learned from the result? then
    if HuskyLens check if ID 1 frame is on screen from the result? then
      serial output join "X:" HuskyLens get X center of ID 1 frame from the result in string , No-Wrap
      serial output join ", Y:" HuskyLens get Y center of ID 1 frame from the result in string , No-Wrap
      serial output join ", W:" HuskyLens get width of ID 1 frame from the result in string , No-Wrap
      serial output join ", H:" HuskyLens get height of ID 1 frame from the result in string , Wrap
    end if
  end if
  wait 1 seconds
end forever

```

Skip to Checking results

Test the object tracking - Option 2 (Arduino UNO/HuskyLens)

Open your Mind+ and load extensions for work with Arduino UNO and HuskyLens camera. Use this code with Arduino/HuskyLens:

```

Uno starts
HuskyLens initialize pin until success
HuskyLens switch algorithm to Object tracking

forever
  HuskyLens request data once and save into the result
  if HuskyLens check if ID 1 is learned from the result? then
    if HuskyLens check if ID 1 frame is on screen from the result? then
      serial output join "X:" HuskyLens get X center of the ID 1 No. 1 frame from the result in string , No-Wrap
      serial output join ", Y:" HuskyLens get Y center of the ID 1 No. 1 frame from the result in string , No-Wrap
      serial output join ", W:" HuskyLens get width of the ID 1 No. 1 frame from the result in string , No-Wrap
      serial output join ", H:" HuskyLens get height of the ID 1 No. 1 frame from the result in string , Wrap
    end if
  end if
  wait 0.5 seconds
end forever

```

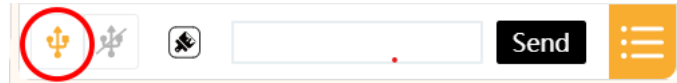


http://erasmus-artie.eu

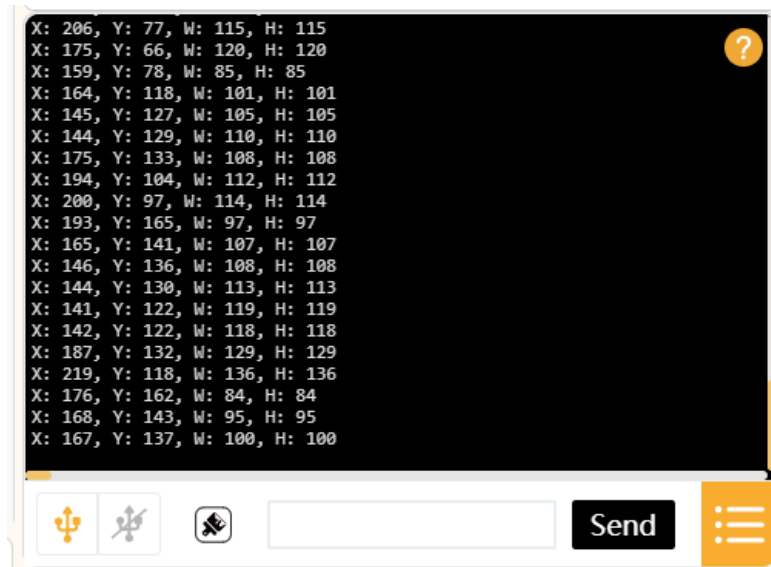


Checking results on serial monitor (both options)

Open serial monitor by clicking on USB icon in lower right part of the Mind+ screen.

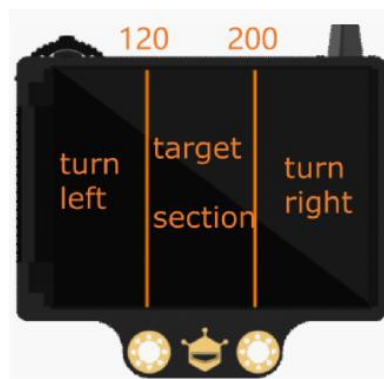


Try to move the object left and right to observe the numerical change of the X centre. Move the object up and down to observe the numerical change of Y centre. Move the object back and forth to observe the numerical change of width and height.



Object motion analysis

As shown in the following picture, the screen is divided into 3 sections according to the X axis of the camera screen coordinate system, and the middle section is our target section.

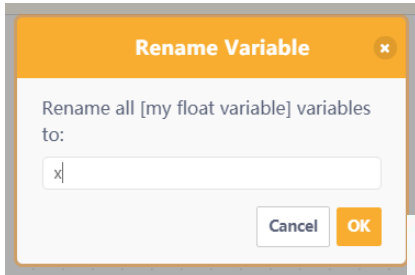




When the camera continuously detects the state of the target object in the picture, its X centre is 120-200, which means that the target is in the centre of the field of vision and robot does not need to adjust its position; its X centre is 0-120, our robot needs to adjust by turning right; its X centre is 200-320, ARTIEbot needs to turn left to adjust.

Now it's time to write the main part of the code to turn the robot toward the object.

Both options - Rename my float variable to x. Right click on variable -> Rename numeric variable.



Option 1 - track the object with Maqueen Plus

Use and configure blocks as in the picture:

```

micro:bit starts
  HuskyLens initialize pin until success
  HuskyLens switch algorithm to Object tracking
  set motor All stop
  forever
    HuskyLens request data once and save into the result
    if HuskyLens check if ID 1 is learned from the result? then
      if HuskyLens check if ID 1 frame is on screen from the result? then
        set x to HuskyLens get X center of ID 1 frame from the result
        if x >= 120 and x <= 200 then
          set motor All stop
        else if x > 0 and x < 120 then
          set motor Left move by 20 speed Forward
          set motor Right move by 80 speed Forward
        else if x > 200 and x < 320 then
          set motor Left move by 80 speed Forward
          set motor Right move by 20 speed Forward
        else
          set motor All stop
      else
        set motor All stop
    else
      set motor All stop
  
```





Option 2 - track the object with ArtieBot

First define blocks **Drive** and **Stop** as described in the previous lesson (Programming the robot)

```
define Drive SpeedA SpeedB Time
  digital pin 8 output LOW
  digital pin 9 output LOW
  if SpeedA > 0 then
    digital pin 12 output HIGH
  else
    digital pin 12 output LOW
  if SpeedB > 0 then
    digital pin 13 output HIGH
  else
    digital pin 13 output LOW
  pwm pin 3 output abs of SpeedA
  pwm pin 11 output abs of SpeedB
  wait Time seconds

define Stop
  pwm pin 3 output 0
  pwm pin 11 output 0
```





se and configure blocks as on picture below:

```

Uno starts
HuskyLens initialize pin until success
HuskyLens switch algorithm to Object tracking
forever
  HuskyLens request data once and save into the result
  if HuskyLens check if ID 1 is learned from the result? then
    if HuskyLens check if ID 1 frame is on screen from the result? then
      set x to HuskyLens get X center of the ID 1 No. 1 frame from the result
      if x >= 120 and x <= 200 then
        Stop
      else if x > 0 and x < 120 then
        Drive 20 80 0.25
      else if x > 200 and x < 320 then
        Drive 80 20 0.25
      else
        Stop
    else
      Stop
  else
    Stop

define Stop
  pwm pin 3 output 0
  pwm pin 11 output 0

define Drive SpeedA SpeedB Time
  digital pin 8 output LOW
  digital pin 9 output LOW
  if SpeedA > 0 then
    digital pin 12 output HIGH
  else
    digital pin 12 output LOW
  if SpeedB > 0 then
    digital pin 13 output HIGH
  else
    digital pin 13 output LOW
  pwm pin 3 output abs of SpeedA
  pwm pin 11 output abs of SpeedB
  wait Time seconds
  
```



<http://erasmus-artie.eu>

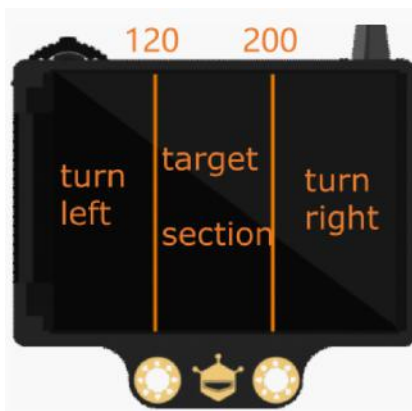
BOTH options - Check how it works

Upload the program to your robot.

Make the adjustments to speed of MotorA or MotorB if necessary.

When the box of identified object is in the centre of the screen, the robot stops.

When the box is on the left or right side of the screen, the robot automatically adjusts the position left or right until the box is located in the target section of the screen.





Follow the object

We got our robots turning to object but still not following. To achieve this, we'll have to detect the size of the object to find out if it's large (close to the camera) or small (far from the camera).

Make a new variable (numeric) and name it h. It will keep the height of the object we are tracking. If the object height is between 60 and 100, the robot will keep the current position. If it's smaller than 60, it's far and we need to make the robot drive forward. If it's higher than 100, the robot should move backward.

Option 1 - Code for Maqueen Plus

```

micro:bit starts
HuskyLens initialize pin until success
HuskyLens switch algorithm to Object tracking
set motor All stop
forever
  HuskyLens request data once and save into the result
  if HuskyLens check if ID 1 is learned from the result? then
    if HuskyLens check if ID 1 frame is on screen from the result? then
      set x to HuskyLens get X center of the ID 1 No. 1 frame from the result
      set h to HuskyLens get height of the ID 1 No. 1 frame from the result
      if x >= 120 and x <= 200 and h >= 60 and h <= 100 then
        set motor All stop
      else if x > 0 and x < 120 then
        set motor Left move by 20 speed Forward
        set motor Right move by 80 speed Forward
      else if x > 200 and x < 320 then
        set motor Right move by 80 speed Forward
        set motor Left move by 20 speed Forward
      else if h > 0 and h < 60 then
        set motor All move by 80 speed Forward
      else if h > 100 and h < 240 then
        set motor All move by 80 speed Backward
    else
      set motor All stop
  
```



<http://erasmus-artie.eu>



Option 2 - Code for Arduino (ArtieBot):

```

Uno starts
HuskyLens initialize pin until success
HuskyLens switch algorithm to Object tracking
forever
  HuskyLens request data once and save into the result
  if HuskyLens check if ID 1 is learned from the result? then
    if HuskyLens check if ID 1 frame is on screen from the result? then
      set x to HuskyLens get X center of the ID 1 No. 1 frame from the result
      set h to HuskyLens get height of the ID 1 No. 1 frame from the result
      if x >= 120 and x <= 200 and h >= 60 and h <= 100 then
        Stop
      else if x > 0 and x < 120 then
        Drive 20 80 0.25
      else if x > 200 and x < 320 then
        Drive 80 20 0.25
      else if h > 0 and h < 60 then
        Drive 80 80 0.25
      else if h > 100 and h < 240 then
        Drive -80 -80 0.25
      else
        Stop
    else
      Stop
  else
    Stop
  
```

```

define Drive SpeedA SpeedB Time
digital pin 8 output LOW
digital pin 9 output LOW
if SpeedA > 0 then
  digital pin 12 output HIGH
else
  digital pin 12 output LOW
if SpeedB > 0 then
  digital pin 13 output HIGH
else
  digital pin 13 output LOW
pwm pin 3 output abs of SpeedA
pwm pin 11 output abs of SpeedB
wait Time seconds

define Stop
pwm pin 3 output 0
pwm pin 11 output 0
  
```

f <http://erasmus-artie.eu>

Both options - Check how it works

Upload the program to micro:bit/Arduino UNO to check how it works.

Do the corrections to make the movement smooth by adjusting motors speed and time of driving.

After HuskyLens finishes learning an object, robots will automatically follow the object and move forward, backward, left and right, keeping the object box in the centre of the screen and at a suitable distance.

When the robot is used as a tracking robot, it can be programmed to locate any target with HuskyLens camera. It means that you can turn this project into a person follower and make the robot follow people.



Object tracking is the task of obtaining a set of values upon object detection, creating a unique identifier for each of the detected objects, and then tracking each of the objects as they move and marking them with a box in the video, respecting the assigned identifiers. State-of-the-art methods include merging data from RGB and other cameras to provide more reliable object tracking.

Now we understand the basic principles of object tracking and have learned the object tracking function of HuskyLens.

We also know how to use HuskyLens so that our robot can track an object.



CONCLUSION

To successfully track an object, we must first detect that object and then track it. Tracking a moving object requires visual object tracking technology as well as manual control.





Learning scenario 17

Duration: 90 min

Topics

Programming robot for line following using a visible light camera, an infrared thermal imager and other detection instruments

Aims

To learn how to program a robot for line following

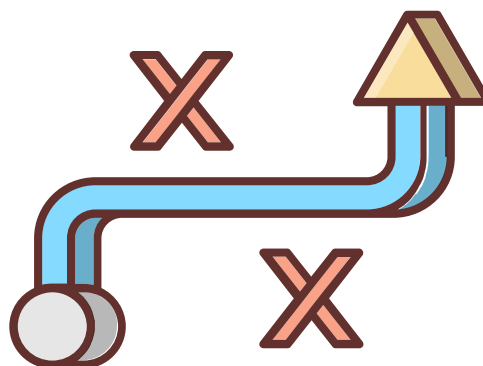
Outcomes

Knowing how to code a robot for line following

Programming a robot - line following

We learned how to program robots to follow objects in the previous lesson.
Make sure your students understand everything from the previous lesson and are ready for the next step.
Let's learn now how to find and follow a line.

Announcement of the goal of the lesson:
How to prepare the robot to follow the line?



<http://erasmus-artie.eu>



MAIN PART

Line tracking refers to the process of object moving along a designated route. A fully functional line tracking robot uses a mobile robot as a carrier, a visible light camera, an infrared thermal imager and other detection instruments as a load system, a multi-field information fusion of machine vision, electromagnetic field, GPS and GIS as a navigation system for autonomous movement and tracking of the robot, and an embedded computer as a software and hardware development platform for the control system.

Line tracking sensors

Type Comparison Items	Infrared Line Tracking Sensor	Visual Sensor
Cost	Low	High
Range of Vision	The sensor has a small range of view; it needs to be close to the ground .	The range of vision is wide, and the movement state can be adjusted in advance according to the line changes.
Environment Adaptation	When the usage environment is changed, the sensitivity of the sensor needs to be adjusted, and the adjustment process is complicated.	When the use environment is changed, only the lines need to be relearned, and the operation is simple.
Map Adaptation	Generally, only suitable for simple maps with clear background lines, black and white lines or solid lines	Suitable for maps with clear background lines, multi-colour lines, solid lines, dotted lines and other complex conditions.



<http://erasmus-artie.eu>

HuskyLens line tracking algorithm

HuskyLens line tracking function is based on Pixy, an open-source project of Carnegie Mellon University. Pixy's algorithm can recognize the colour of pictures. Its basic idea is to use the colour space to remove the background no users are interested in and extract the foreground (such as lines).

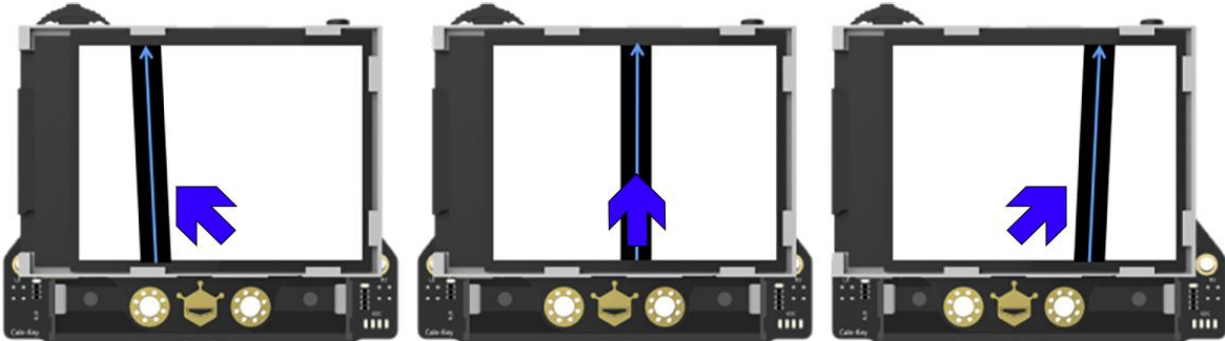


How can robot follow the black line on the tracking map (which has black lines on white ground)? In fact, we only need to know the relative position of ARTIEbot to the black line.



Basically, we have following three situations:

1. When robot is on the right side of the black line, it should turn left
2. When robot is in centre, aligned with the black line, it should go straight
3. When robot is on the left side of the black line, it should turn right

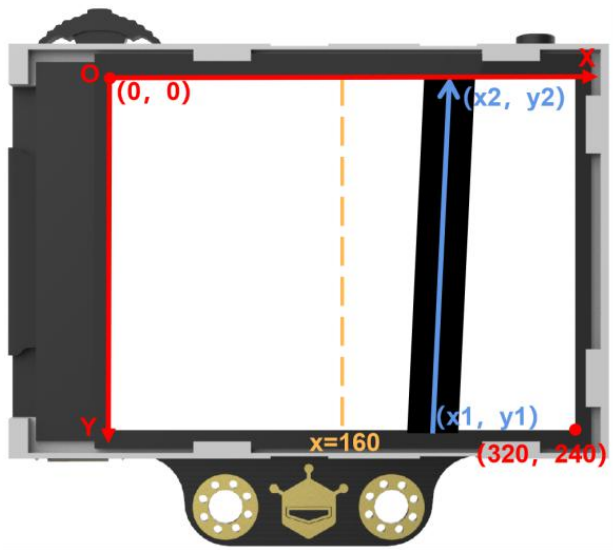


Implementation

The resolution of HuskyLens screen is 320×240. The O point in the upper left corner of the screen is the origin of the screen's coordinates (0, 0), the horizontal right direction is the positive direction of the X axis, and the vertical down direction is the positive direction of the Y axis, so the coordinates in the lower right corner of the screen are (320, 240). The dotted orange line in the picture is the central axis of the screen, and the x value of this line is 160. The black line in the screen below is the map line "seen" by HuskyLens camera. The blue arrow is the line direction calculated by HuskyLens. The starting point coordinates of the blue arrow are (x1, y1) and the ending point coordinates are (x2, y2).



<http://erasmus-artie.eu>





To make it simple - we only need to know the starting point (x1) of the blue arrow relative to the central axis (x=160) to implement line tracking.

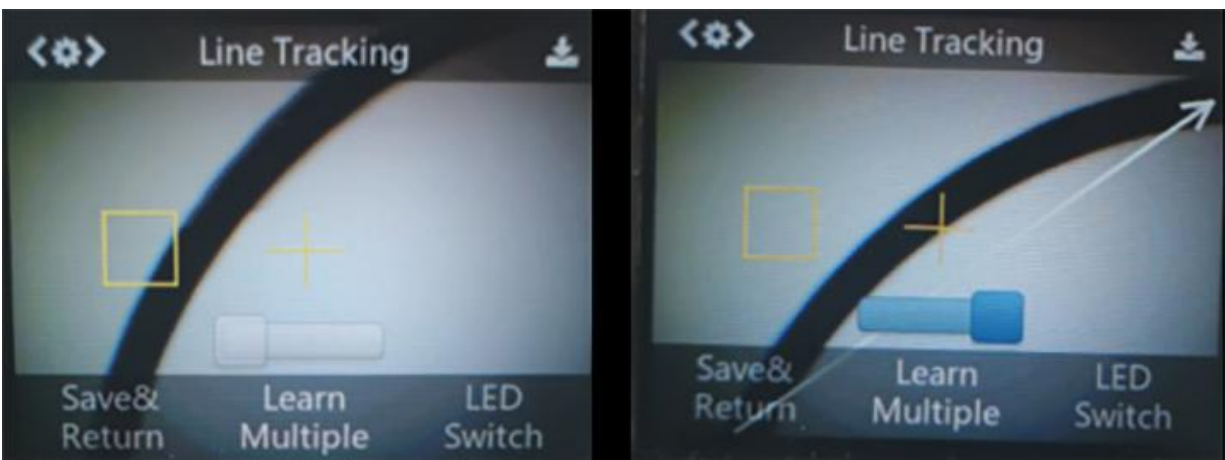
This function can track lines of specified colours and make path prediction. The default setting is to track lines of only one colour and this project will use one colour line tracking

Camera Settings

Step 1: Dial the function button to the left or right until the word "Line Tracking" is displayed at the top of the screen.

Step 2: Long press the function button to enter the parameter setting of the line tracking function.

Step 3: Dial the function button right or left until "Learn Multiple" is selected, then short press the function button, and dial it to the left to turn off the "Learn Multiple" switch, that is, the square icon on the progress bar is turned to the left. Then short press the function button to complete this parameter.

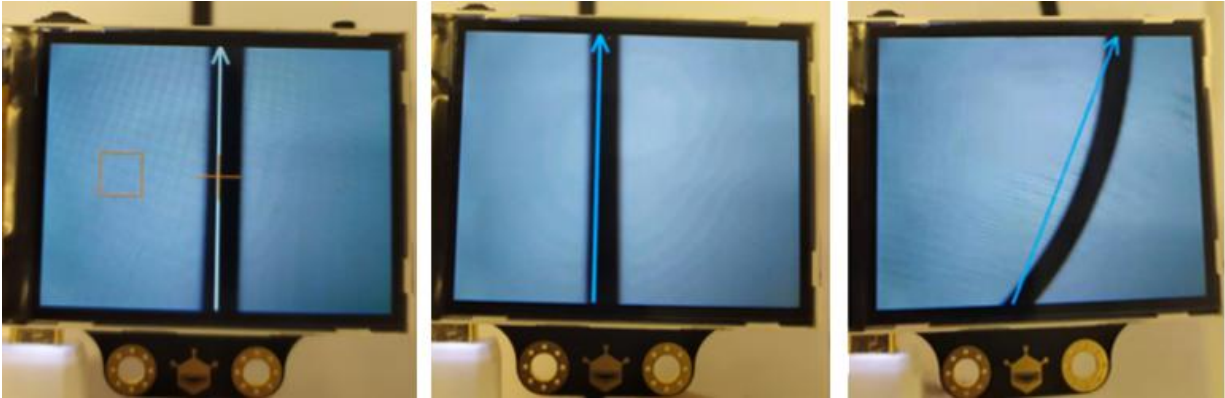


Step 4: You can also turn on the LED by setting "LED Switch". This is very useful in the dark environment. Referring to the above method, turn on the "LED Switch".

Step 5: Dial the function button to the left until "Save & Return" is selected, and short press the function button to save the parameters and it will return automatically.

Learning and Tracking

Line Learning: Point the "+" symbol at the line, then point the orange box at the background area. It is recommended that no other lines are on the screen. Try to keep HuskyLens parallel to the target line; HuskyLens will automatically detect the line and a white arrow will appear. Then short press the "learning button" and the white arrow turns into a blue arrow.



- When learning the line, we need to adjust the position of HuskyLens to be parallel to the line.
- HuskyLens can learn line in any colour that have an obvious colour contrast to background, but this line must be monochrome to keep line tracking process stable.
- HuskyLens can learn and track multiple lines with different line colours, but any of these lines must be monochromatic and in visible contrast against the background. In this example we will use black line (black insulation tape on white background such is paper or white MDF).
- The visibility of the line depends much on the ambient light. When following the line, please try to keep the ambient light as stable as possible and use HuskyLens LED if necessary.



Open your Mind+ and load extensions.

Open your Mind+ and load extensions. **When using the micro:Maqueen plus robot, make sure that you select the right version (V1 or V2).**

Rename **my float variable** to **x**. Right click on variable -> Rename numeric variable.

Algorithm

- Read x_1 (or x beginning) from HuskyLens Line tracking function - it is the beginning point of the blue arrow
- If the black line is on the left side of the screen ($x_1 < 150$) - robot should turn left.
- If the black line is on the right side of the screen ($x_1 > 170$) - robot should turn right.
- If the black line is in the middle of the screen ($150 \leq x_1 \leq 170$) - robot should go straight.



Option 1 - Line following with Maqueen Plus

Use this code:

```
micro:bit starts
HuskyLens initialize pin until success
HuskyLens switch algorithm to Line tracking
forever
  HuskyLens request data once and save into the result
  if HuskyLens check if ID 1 is learned from the result? then
    if HuskyLens check if ID 1 arrow is on screen from the result? then
      set my float variable to HuskyLens get X beginning of ID 1 arrow from the result
      if my float variable >= 150 and my float variable <= 170 then
        set motor All move by 60 speed Forward
      else if my float variable < 150 then
        set motor Right move by 90 speed Forward
        set motor Left move by 30 speed Forward
      else if my float variable > 170 then
        set motor Left move by 90 speed Forward
        set motor Right move by 30 speed Forward
      else
        set motor All stop
```





Option 2 - Line following with Arduino (ArtieBot)

First define blocks Drive and Stop as described in lesson Programming the robot. It is a good idea to put the often-used scripts in backpack and easily move them between projects.

```
define Drive SpeedA SpeedB Time
  digital pin 8 output LOW
  digital pin 9 output LOW
  if SpeedA > 0 then
    digital pin 12 output HIGH
  else
    digital pin 12 output LOW
  if SpeedB > 0 then
    digital pin 13 output HIGH
  else
    digital pin 13 output LOW
  pwm pin 3 output abs of SpeedA
  pwm pin 11 output abs of SpeedB
  wait Time seconds

define Stop
  pwm pin 3 output 0
  pwm pin 11 output 0
  digital pin 8 output HIGH
  digital pin 9 output HIGH
```





```

Uno starts
HuskyLens initialize pin until success
HuskyLens switch algorithm to Line tracking
forever
  HuskyLens request data once and save into the result
  if HuskyLens check if ID 1 is learned from the result? then
    if HuskyLens check if ID 1 arrow is on screen from the result? then
      set x to HuskyLens get X beginning of ID 1 arrow from the result
      if x >= 150 and x <= 170 then
        Drive 90 90 0.25
      else if x < 150 then -
        Drive 40 90 0.25
      else if x > 170 then -
        Drive 90 40 0.25
      else
        Stop
  
```

```

define Drive SpeedA SpeedB Time
  digital pin 8 output LOW
  digital pin 9 output LOW
  if SpeedA > 0 then
    digital pin 12 output HIGH
  else
    digital pin 12 output LOW
  if SpeedB > 0 then
    digital pin 13 output HIGH
  else
    digital pin 13 output LOW
  pwm pin 3 output abs of SpeedA
  pwm pin 11 output abs of SpeedB
  wait Time seconds
  
```

f

<http://erasmus-artie.eu>



Both options - Test your algorithm

Prepare the white surface (paper or MDF) and use the insulation tape to make a line (see example below or make something similar).



Upload the program to your robot, place the robot somewhere on the line and see how it moves. Does it follow the line?

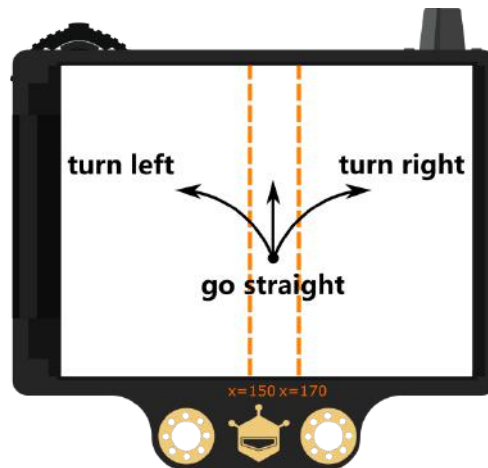
Is the line being lost? Is there a way to deal with it? The following hints will help you:

- Try changing the angle of camera
- Try to get **X beginning** of ID 1 arrow instead of X endpoint

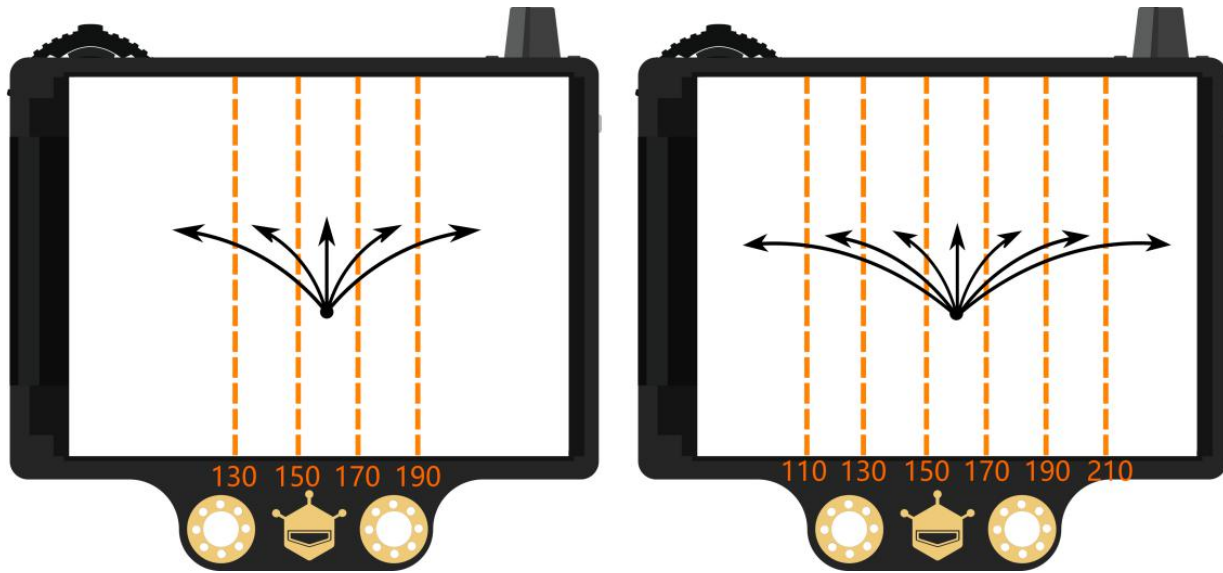
Add fail-safe code using following hint:

- If a line is present and an arrow is detected, proceed to the arrow handling code and save the content of x variable to a new variable called lastx
- If the line is lost and the arrow is not detected, use lastx instead of x to reach the line again.

And finally the most important - apply the corrections to make the movement smooth by adjusting motors speed and time of driving. This algorithm analyses the position of x; it has 3 cases as in the picture below.



Can you optimise this algorithm for 5 cases or even 7 to make the movement smoother? See pictures below to get the idea how to do this.



Knowledge Recap

1. Understand the main principles of line tracking
2. Learn to apply HuskyLens line tracking function
3. Apply and optimise the line following algorithm

As the name suggests, the line follower robot is an automated vehicle that follows a visual line embedded on the surface. This visual line is a path on which the line follower robot runs. Generally, it uses a black line on a white surface, or you can adjust it as a white line on a black surface.

Usually, beginners and students would get their first robotic experience with this type of robot. In industries, giant line follower robots are used for assisting the automated production process. They are also used in military applications, human assistance purposes, delivery services, etc. Now we understand the main principles of line tracking, and know how to apply HuskyLens line tracking function.

We also know how to apply and optimize the line following algorithm.

Discuss with your students the differences and similarities between object tracking and following and line tracking and following.

CONCLUSION

We learned the main principles of line tracking using the HuskyLens function and optimized the line tracking algorithm.