

## Robot eyes - Intro to TPBot and Smart AI lens

In this lesson, students will be working with a TPBot smart car, a BBC micro:bit, with an AI Lens that enables the robot to 'see' and recognise colours using simple image recognition.

Students begin by exploring a sample program that allows the TPBot smart car to respond to red and blue colour cards. They will predict what the robot will do, run the program, and investigate how the AI lens works. They will modify the code to include new behaviours (such as detecting a third colour or adding sound or movement), before finally making their own version of a smart colour-responding robot.

This hands-on, real-world learning task supports students in designing an algorithm, interpreting sensor input, and understanding AI decision-making – all within the context of a fun, creative robotics challenge.

### Curriculum links

This activity and the suggested design challenges are aligned with elements of the following learning area. We encourage teachers to adapt the content and focus to suit the needs of their students. They can be adapted for students between **Year 3 and Year 6**. More details are provided in the supporting documents.

#### Digital Technologies

### Required resources

You will need the following materials which are available in our CSER lending library kit:

- BBC micro:bit V2
- TPBot smart car
- PlanetX AI Smart Lens sensor module
- Red, blue, and green colour cards
- USB cable and battery pack
- Laptop or tablet with Microsoft [MakeCode](#) editor



### Pre-requisite knowledge

Prior to this session, students should:

- know basic programming structures: sequences, simple loops, and conditionals (if statements).
- have basic experience using MakeCode or block-based editors.
- understand how to download code to a micro:bit.

Technologies Core concepts	Systems – inputs, processes and outputs	Interactions and impacts
	Systems thinking	Computational thinking
For more information about the Australian Curriculum Technologies core concepts <a href="#">Understand this learning area - Technologies</a>		

## Suggested steps

This activity will follow the PRIMM pedagogical framework. The PRIMM framework is a structured approach to teaching computer programming that emphasises reading and understanding existing code before writing new programs. It consists of five stages: Predict, Run, Investigate, Modify, and Make. This scaffolded method aims to reduce cognitive load, promote discussion, and gradually build students' confidence and ownership in programming.

### PRIMM Programming pedagogy

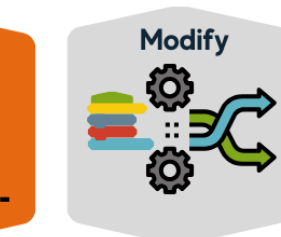
What does the code do?  
Make predictions



Explore each line of the code,  
trace and comment



Run the code and  
test predictions



Adapt and change the code  
to extend its function

Plan and code a  
new program



S4S Coding <https://s4scoding.com/national-curriculum-in-england-computing-programmes-of-study/?cid=5>

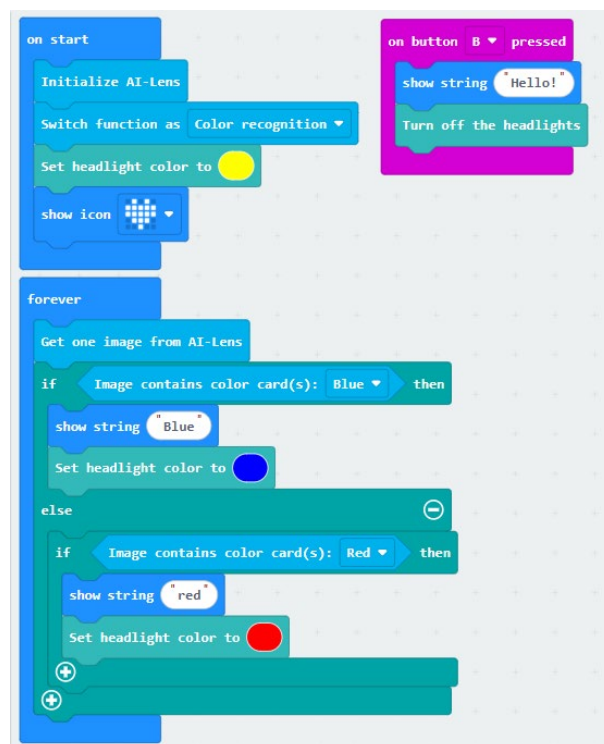
### 1. Predict

Present the following code to students (either through [LINK](#) on a shared screen or in printed format available in Supporting documents).

Ask them to **Predict** what each block will achieve. Describe what the code does, in individual steps by identifying elements of the algorithm such as:

- What happens when it starts?  
(Initialises the AI Lens ready to identify a colour, sets TPBot headlight to yellow and shows a heart icon on the micro:bit LED display)
- What are the inputs required for an action to occur?  
(Show the AI lens a colour card or Press Button B)
- What decisions are made?  
(Is the colour card blue or red)
- Can they identify any repetitions or loops?  
(The AI Lens will repeatedly check the lens to identify blue or red)

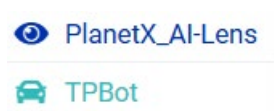
- What do you think will happen when the robot sees a red card?
- What happens if no colour is detected?
- What might Button B be used for in this program?




## 2. Run

Either provide students with the following [link](#) to the MakeCode project OR have students create their own by copying from a printed version provided (see Supporting documents below). Please note, if creating a new project, students will need to add the extensions for the TPBot and AI lens to their MakeCode editor.

**Click +Extensions** and select the following:



Students upload the code to the micro:bit that is then connected to the TPBot. and attach the AI Lens to the TP Bot and connect to the port and run the program.

 An error message “initialise error” will appear in the simulator and on the micro:bit because the micro:bit is attempting to detect the AI lens. Once the micro:bit is inserted into the TPBot and the lens is properly connected, the system initialises correctly and the error message is resolved on the physical micro:bit but will continue on the simulator. If it continues on the micro:bit once connected to the TP-Bot, press the RESET button on the back of the micro:bit.

Students hold red and blue cards in front of the robot and observe its behaviour. Trial and error is required to determine the best position of the card in relation to the camera. They could test the code using red or blue objects and discuss their findings.

- Were their predictions correct?
- What did they notice?



## 3. Investigate

Guide students through identifying key parts of the code:

- Where is the AI lens initialised?  
(On start the initialised block sets up the AI lens so that it's ready to work with the robot and receive image data. The switch function block tells the AI lens to switch to colour detection mode so that it will start looking for colours in the camera view.)
- How does the robot detect colour?  
(The block <if image contains colour card(s) blue> checks if blue is detected in the camera's view. The AI lens camera takes an image and checks that image for specific colours like blue or red. If the result is true, the robot runs the next block of code. This is repeated for each colour it needs to check)
- What happens in the 'if and else if' sections of the code?  
(These blocks tell the robot what to do when it sees a certain colour. For example, if the colour blue is detected, then the robot displays the word "blue" on the LEDs and changes the TPBots headlights to blue. Then the next block is checked if blue is not found, but red it then the robot displays the word "red" on the micro:bit screen and changes the headlights to red.)

## 4. Modify

Have students make changes to the code. Suggested challenges:

- Add a new colour condition (e.g., green or yellow).
- Replace **show String block with different icons** or words
- Add movement commands to make the robot move forward, backward, or turn.
- Use music blocks to make the robot play a sound when it detects each colour.

## 5. Make (See Supporting documents for more ideas to extend this challenge)

Students design their own "colour command robot" maze or challenge. We have included some example problem statements and user stories in the Supporting document section that could be used to further develop a classroom activity.

- Use coloured markers or cards as inputs.
- Create a path that uses AI colour detection to make decisions.

## Pose questions

Use these example open-ended questions to prompt student reflection, check for understanding, and encourage discussion to help students explain their thinking and build on each other's ideas.

- What changes did you make to the original code?
- Did anything go wrong when you were testing? How did you fix it?
- What was the most difficult part of changing the code?
- What advice would you give another group starting this activity?
- Can you think of a situation where a robot might need to make decisions based on colour?



## Why is this relevant? (Real world connections)

This activity introduces students to **AI-powered decision making**, an important part of many real-world technologies, such as:

- Self-driving cars recognising traffic signals and signs
- Robots sorting recyclable materials based on colour
- Agricultural robots detecting ripeness of fruit
- Smart devices interpreting visual data for accessibility

Students gain insight into how **AI uses inputs (like colour or images)** to make decisions – a foundational concept in emerging digital technologies.

## Assessment

Observation can be used to check students' ability to carry out tasks aligned to the Australian Curriculum. We have included some suggested questions for teachers to reflect on and to guide these observations.

### Checking for understanding

- Were students able to describe the steps in the code in their own words?
- Did they correctly identify where decisions (branching) happened in the program?
- Were they able to debug their code if the robot didn't behave as expected?
- Did they successfully modify the code? Could they explain the code block used?

For more assessment resources we recommend the Assessment resources on the [Digital Technologies Hub](#).

## Teacher professional learning opportunities

We would like to thank the Australian Government Department of Education for funding our Lending Library and associated resource development.



We run a range of STEM programs for Australian teachers, including our online CSER MOOC courses, free professional learning events, and our National Lending Library.

Our free, self-paced online courses available from CSER and Maths in Schools in the following areas:

- Decoding Digital Technologies
- Digital Technologies + X
- Cyber Security and Awareness
- Teaching AI in the classroom
- Maths in Schools: Foundation - Year 2, Year 3 - 6 and Year 7 - 10

[www.csermoocs.adelaide.edu.au](http://www.csermoocs.adelaide.edu.au)



## Robot eyes - Intro to TPBot and AI lens Supporting document - Year 5 & 6 design challenges

After completing the previous steps, students in Year 5 & 6 can complete a design challenge which completes the PRIMM framework in more detail with the 'Make' stage.

### Design your own colour command robot

Now that students have predicted, tested, explored, and modified the code, it's time to **design their own smart robot** system. The challenge is to **create a custom version** of the colour-detecting TPBot that can perform actions based on the colours it "sees." This is a chance to combine coding with creativity — just like real engineers and AI designers.

We have included some example problem statements and user stories that could be used to further develop a classroom design activity.

---

### Design ideas

---

#### Maze navigation

Create a simple maze using coloured cards at decision points. Each colour tells the TPBot what to do: e.g., blue = forward, red = reverse, green = turn left, yellow = stop.

- Problem statement - In a warehouse, small delivery robots need to follow colour-coded signs to reach the correct loading zone without crashing or getting lost.
- User story - As a warehouse manager, I want the robot to move forward when it sees blue and turn left at green, so that it can follow a delivery route without needing human control.

#### Delivery bot

Set up coloured "delivery zones" around the room. Use a map or path and program the robot to go to a zone based on the colour you show.

- Problem statement - In a large library or office, small delivery robots are needed to bring supplies to different coloured zones (e.g., red = printing, blue = tech support). The robot must recognise the coloured markers and deliver items without human direction.
- User story - As a library assistant, I want the robot to detect coloured zones and move in different directions, so that it can deliver books or supplies to the correct area quickly and safely.

#### Dance routine

Use coloured cards to trigger robot dance moves (turns, flashing lights, icons, etc.). Create a short routine using timed colour inputs.

- Problem statement - A school performance group wants to include a robot in their show. The robot should perform a short dance routine by responding to colour cards held up by the audience or teacher. Each colour triggers a specific move, light pattern, or sound.
- User story - As a school performer, I want the robot to respond to colour cards with fun movements and lights, so that it can be part of our performance and entertain the audience.





## Emergency bot

Imagine your robot is a rescue vehicle. Each colour represents a different task (e.g., blue = water spill, red = fire hazard). Create a simulation.

- Problem statement - In emergency drills, a robot could be used to guide students by responding to colour-coded signs (e.g., green = go, red = danger). It needs to detect and respond to these colours to help lead students to safety during practice drills.
- User story - As a school safety officer, I want the robot to recognise emergency colour signals and respond, so that it can help guide students during safety drills or alerts.

## User stories

User stories are brief descriptions of the needs and goals of the users of a product or service. They help define the features and functionalities that will create user value and solve their problems. A suggested format would look like this:

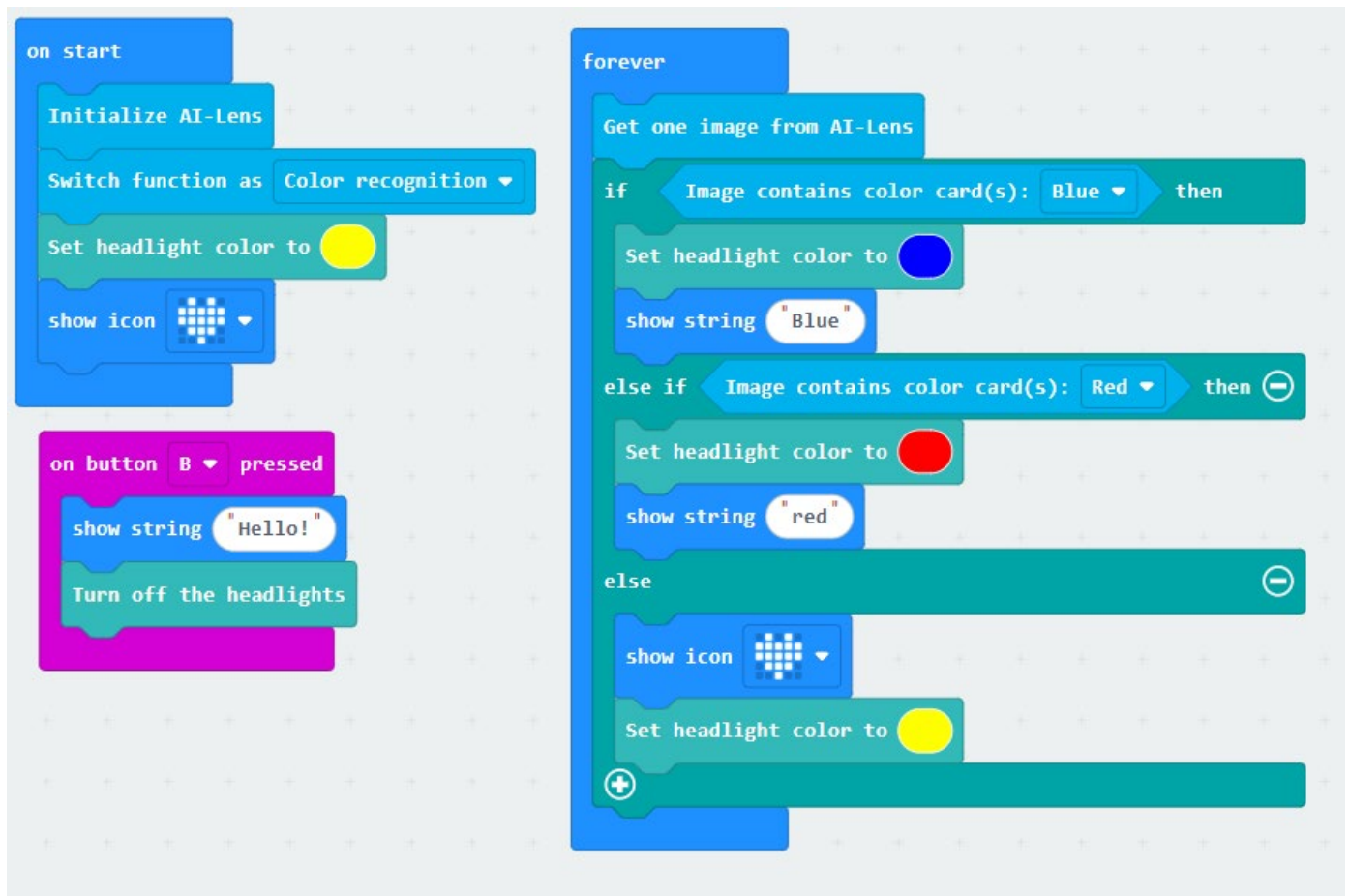
A **< insert a user >** has **< insert a goal >** so that **< insert a reason >**  
for whom? do what? why?

## Problem statements

Describing the problem or opportunity is important as it ensures the purpose and goals are explicit. A number of structures can be used to support students in creating good statements including the following.

How might we **< insert action >** **do what?**  
for **< insert user >** **for whom?**  
in order to **< insert goal, benefit, gain > ?** **why?**

## Sample MakeCode project – TPBot and AI lens



To access project

<https://makecode.microbit.org/S83762-80371-95243-02789>







## Australian Curriculum

This activity is suitable for students between Year 3 and Year 4 **Digital Technologies**:

Students in Years 3 and 4 learn to:

- follow and describe algorithms involving sequencing, comparison operators (branching) and iteration (AC9TDI4P02)
- implement simple algorithms as visual programs involving control structures and input (AC9TDI4P04).

By the end of Year 4 students follow and describe simple algorithms involving branching and iteration and implement them as visual programs.

**If adapted to include a design challenge, the activity would align with the following curriculum content**

Students in Years 5 and 6 learn to:

- define problems with given or co-developed design criteria and by creating user stories (AC9TDI6P01)
- generate, modify, communicate and evaluate designs (AC9TDI6P04)
- evaluate existing and student solutions against the design criteria and user stories and their broader community impact (AC9TDI6P06)
- design algorithms involving multiple alternatives (branching) and iteration (AC9TDI6P02)
- implement algorithms as visual programs involving control structures, variables and input (AC9TDI6P05).

By the end of Year 6 students develop and modify digital solutions, and define problems and evaluate solutions using user stories and design criteria. They design algorithms involving complex branching and iteration and implement them as visual programs including variables.



This work is licenced under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License. [CC BY-NC-SA](https://creativecommons.org/licenses/by-nc-sa/4.0/)  
Computer Science Education Research (CSER) Group, The University of Adelaide