

Introduction to Programming

LEGO® MINDSTORMS® EV3

Teacher's Guide

*The Introduction to Programming EV3 Curriculum was
produced by Carnegie Mellon's Robotics Academy*

The Introduction to Programming EV3 Curriculum is not a LEGO® MINDSTORMS® product.
LEGO Education or the LEGO Group does not sponsor, endorse, or support this product.

Preface

There is a growing recognition that *Computational Thinking Practices* are critical for all students to learn. They form the cornerstone of the language of innovation, and will drive all future STEM discoveries. They are a new set of “basic skills” that all students need to know.

But what are they? At first glance, concepts like “consider problems analytically” and “use data to inform decisions” seem abstract and difficult to comprehend. Educational robotics systems like the EV3 provide a much-needed tool to make them real and approachable.

Consider the first few activities in this curriculum: students program a robot to drive fixed distances in set patterns. Even these simple programming constructs require precise, thoughtful communication between student and robot – how far should the robot move? How far should it turn? As the challenges become more complex, students learn to break the large problems down into simpler ones, and construct solutions with care, one step at a time.

Sensors add the element of data and make key information about the robot’s environment available; numeric abstractions become a reality – 35 centimeters to the nearest wall, turn 90 degrees – enable the student to make smart decisions about the robot’s behavior.

These practices – precise logical thinking, using data to make decisions, analyzing problems, and building solutions in teams – are critical in all forms of problem-solving, not just robotic ones.

Robotics activities are concrete, contextualized, and provide immediate feedback – important factors in satisfying a student’s desire for success and creating the motivation to continue learning. Students also learn about the robotics technologies themselves, which impact all modern industries, from agriculture to healthcare, banking, manufacturing, transportation, energy, and security. The pervasiveness of robotics technologies, from airplane autopilots, to bank machines, to smartphones, to self-driving cars helps students to be “engaged learners” as they believe that the content that they are studying is important or will be valuable to them.

The Introduction to Programming curriculum is just that: an introduction. For many teachers this will be your first experience at teaching robotics and programming. If you need help, the Robotics Academy has lots of free resources on its website and regularly offers teacher courses. If you have questions or find issues, we would love to hear from you.

Enjoy your school year.

Ross Higashi

Ross Higashi,
Robotics Academy Learning Scientist



Robin Shoop,
Robotics Academy Director





Table of Contents

- 3 FAQ
- 4 Checklist

1 Introduction

- 5 What is the Introduction to Programming EV3 Curriculum?
- 6 Why should I use the Introduction to Programming EV3 Curriculum?
- 6 What are the Curriculum's Learning Objectives?
- 6 When should I use the EV3 Curriculum in my class?
- 7 How do I use the Curriculum?
- 8 What topics are covered in each Unit?
- 9 What "Big Ideas" does the Curriculum teach?

10 Standards

- 10 Math Practices
- 10 Math Content
- 11 Common Core English/Language Arts
- 11 Next Gen Science Standards
- 12 Computer Science Standards

13 Classroom Setup

- How should student work stations be setup?
- What are the System Requirements?

14 In the classroom

- 14 General layout of all Units
- 14-17 Batteries, Firmware, Ports, Menus
- 18 -19 Big Ideas that all students will learn
- 20 Using the EV3 Software

21 The Movement Unit

- 21-28 The Moving Straight Chapter
- 29-35 The Turning Chapter

36 The Sensor Unit

- 36-42 The Touch Sensor Chapter
- 43-49 The Ultrasonic Sensor Chapter
- 50-55 The Gyro Sensor Chapter
- 56-62 The Color Sensor Chapter

63 The Robot Decisions Unit

- 63-69 The Loop Chapter
- 70-76 The Switch Chapter
- 77-82 The Switch Loops Chapter
- 83-86 The Line Follower Chapter

87 The Final Challenge

- 87-89 Final Challenge Resources
- 90-91 The Search and Rescue Challenge

92 Reproducibles

Pages 92 - 142 Unit Quizzes, Answer Keys, Handouts, Worksheets and Rubrics

Note: We have been asked by practicing teachers NOT to make these public and so they are not printed with this document. They are made available with the curriculum.





Frequently Asked Questions (FAQ)

Before starting

- ▶ Will *Introduction to Programming* help me teach to Standards?
Yes! See Standards, pages 10 - 12.
- ▶ What do I need to prepare for class?
See Checklist, page 4 and Best Workstation Setup page 15.
- ▶ What topics are covered?
See Topics, page 8 and General Layout, page 14.
- ▶ What's the lesson structure?
See How To Use, page 5.
- ▶ I already have programming tutorials in my software. Is this the same thing?
No. *Introduction to Programming* focuses on building critical thinking skills through programming, rather than rote knowledge of code. See What are the Big Ideas taught..., page 9.

During class

- ▶ How do I begin with *Introduction to Programming* in my class?
See How do I use the *Introduction to Programming* Curriculum in my classroom, page 7.
- ▶ Are there notes available to help me teach the lessons?
Yes. Every page in *Introduction to Programming* is summarized and annotated starting on page 14. There are additional notes at the beginning of each chapter.
- ▶ What do I do about students who go faster/slower than the others?
All lessons are self-paced, so minor variation in pacing is not a problem. You can also include or omit activities marked as Optional, and even let students work ahead on later chapters.

After class

- ▶ Are there quizzes or homework?
Each Chapter includes one or more Reflection Questions designed to let students apply their skills and knowledge to a more sophisticated and writing-intensive task.
You can find additional Handouts, Worksheets and Rubrics starting on page 92.





Checklist

Identify the Focus of your Lesson

Robotics can be used to teach lots of standards. This curriculum is designed to introduce students to how to program, an important part of robotics, but not the only thing that you can teach through robotics. Please read pages 5 - 12 of this guide to learn more.

Set up the student workstations

See page 13, Workstation Setup.

(Recommended) Build the Driving Base for each robot

Since mechanisms aren't the focus of this module, pre-building the basic robot for your students can save multiple weeks of class time and allow them to begin work immediately on Day 1. The plans can be found in the Moving Straight Unit page 23.

Become familiar with the lessons

See page 7 to become familiar with the lesson flow. The general layout of the Introduction to Programming the EV3 Curriculum is found on page 14. Review the first couple lessons starting with Moving Straight on page 21.

Determine overall pacing for the module

Identify key dates that you would like to have each project due by; make these clear to students in your syllabus or assignment sheets.

Review Big Ideas and Computational Thinking

See pages 18 and 19.

Review and print the Reflection Questions for each chapter

Chapter review questions, answer guides, and rubrics begin on page 92.

Note: The reflection questions can be used as class discussion questions, given as homework, or as a quiz.





What is the *Introduction to Programming EV3 Curriculum*?

10 Projects and One Capstone Programming Challenge

The *Introduction to Programming EV3 Curriculum* is a curriculum module designed to teach core computer programming logic and reasoning skills using a robotics engineering context. It contains a sequence of 10 projects (plus one capstone challenge) organized around key robotics and programming concepts.

Each project comprises a self-contained instructional unit in the sequence, and provides students with:

- ▶ **An introduction to a real-world robot** and the context in which it operates
- ▶ **A challenge** that the robot faces
- ▶ **A LEGO-scale version of the problem** for students to solve with their robots
- ▶ **Step-by-step guided video instruction** that introduces key lesson concepts (e.g. Loops) by building simple programs that progress toward the challenge task
- ▶ **Built-in questions** that give students instant feedback on whether they understood each step correctly, to aid in reflection and self-pacing
- ▶ **Semi-guided “Try It!” exploration activities** that expose additional uses for and variants on each behavior
- ▶ **Semi-open-ended Mini-Challenges** which ask students to use the skill they have just learned to solve a relevant small portion of the final challenge
- ▶ **The Unit Challenge** based on the original robot’s problem, for students to solve in teams as an exercise and demonstration of their mastery of the concept
- ▶ Additional **Reflection Questions** found in the back of this Teacher’s Guide allow you to assess the depth of students’ understandings while challenging them to apply their learning to a higher-order problem-solving and writing task.





Introduction to Programming LEGO® MINDSTORMS® EV3

Why should I use the Introduction to Programming EV3 Curriculum?

Introduction to Programming provides a structured sequence of programming activities in real-world project-based contexts. The projects are designed to get students thinking about the patterns and structure of not just robotics, but also programming and problem-solving more generally.

By the end of the curriculum, students should be better thinkers, not just coders.

What are the Learning Objectives of the Introduction to Programming EV3 Curriculum?

- ▶ Basic concepts of programming
 - Commands
 - Sequences of commands
- ▶ Intermediate concepts of programming
 - Program Flow Model
 - Simple (Wait For) Sensor behaviors
 - Decision-Making Structures
 - Loops
 - Switches
- ▶ Engineering practices
 - Building solutions to real-world problems
 - Problem-solving strategies
 - Teamwork

When should I use the Introduction to Programming EV3 Curriculum with my class?

Introduction to Programming the EV3 is well-suited for use at the beginning of a robotics class, as it will allow students to engage immediately and begin building core programming and problem-solving skills before undertaking more ambitious open-ended projects later in the course. This curriculum module should take approximately 6 weeks.





Introduction to Programming LEGO® MINDSTORMS® EV3

How do I use the Introduction to Programming EV3 Curriculum in my class?

Introduction to Programming is designed for student self-pacing in small groups, preferably pairs. Each pair of students should work together at one computer, with one EV3 robot.

Curriculum tasks are designed to involve some – but not extensive – mechanical consideration, so that hands-on design tasks may remain authentic without becoming logistically difficult.

Solutions will not require parts in excess of those included in the 45544 EV3 Core set, so it is sufficient to leave each team with one kit (although access to additional parts may allow students to construct more creative solutions to problems).

A typical plan for an Introduction to Programming chapter is:

1. View the introductory video as a class, or in individual groups, then review the challenge task for the unit
 - In a group, identify and note key capabilities the robot must develop, and problems that must be solved in individual engineering journals or class logs (e.g. on sticky paper posted on the walls)
2. Groups proceed through the video trainer materials at their own pace, following the video instruction directly, and constructing solutions to the Try It! and Mini-Challenge steps as they go
3. Each group constructs its own solution to the Unit Challenge
 - Groups may be asked to document their solutions in journals or logs, and especially to explain how they overcame the key problems identified at the start of the unit
4. Assign the Reflective Question for the chapter
 - Students answer the Reflection Question for the chapter individually, as an in-class or homework assignment
 - Reflection Questions for each chapter can be found in the Reproducibles section of this Teacher's Guide





Introduction to Programming LEGO® MINDSTORMS® EV3

What topics are covered in each Unit?

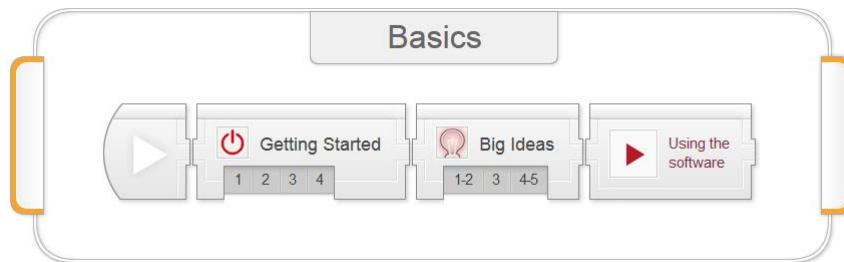
Unit Name	Main Topics
1. Moving Straight	Motors, Sequences of Commands, Block Settings, Downloading and Running Programs, Move Steering Block
2. Turning	Turning, Types of Turns, Move Steering vs. Move Tank Block
3. Move Until Touch	Sensors, Wait For Block, Touch Sensor, Move Until Behaviors
4. Move Until Near	Ultrasonic Sensor, Thresholds
5. Turn for Angle	Gyro Sensor, Compensating for Sensor Error
6. Move until Color	Color Sensor
7. Loops	Loops, Patterns of Behavior
8. Switches	Switches, Conditional Reasoning
9. Switch-Loops	Obstacle Detection Behavior, Repeated Decisions Pattern
10. Line Follower (Mini-Unit)	Line Following (a Repeated Decisions Pattern Behavior)
11. Final Challenge	Cumulative Application of Skills and Knowledge





Introduction to Programming LEGO® MINDSTORMS® EV3

What are the Big Ideas taught in the Introduction to Programming EV3 Curriculum?



Robotics can be something you teach *with*, as well as something you teach *about*. *Introduction to Programming* uses robots, and covers robotics content, but ultimately seeks to give students experience and access to a much broader set of skills and perspectives called Computational Thinking.

► **Big Idea #1: Programming is Precise**

If you want a robot to do something, you need to communicate that idea with mathematical and logical precision, or it won't quite be what you intended.

► **Big Idea #2: Sensors, Programs, and Actions**

Data from sensors gives a robot information about its environment. A program uses that data to make decisions, and the robot Acts on those decisions. Data underlies the core of the entire process.

► **Big Idea #3: Make Sense of Systems**

To understand the way something works, construct a mental "model" of it in your head that captures the important features and rules of the system. This helps you make sense of it, and also gives you a tool to "play out" (similar) new scenarios in your head to predict what would happen.

► **Big Idea #4: Break Down Problems and Build Up Solutions**

To solve a difficult problem, try breaking it down into smaller problems. Then, solve the smaller problems, building up toward a solution to the big problem.

► **Big Idea #5: Computational Thinking Applies Everywhere**

These skills – mathematical and logical clarity, using data, systems thinking with mental models, and problem solving – are not just for robotics. They are key to solving many problems in the world.

A video introduction to these topics can be found in the "Big Ideas" block of the Basics section of the product.





Introduction to Programming LEGO® MINDSTORMS® EV3

What Standards does the Introduction to Programming EV3 Curriculum address?

Common Core Mathematics Practices

Skills math educators at all levels should seek to develop in their students

Standard (CCSS.Math.Practice)	Introduction to Programming the EV3
MP1 Make sense of problems and persevere in solving them	Chapters are all based around solving real-world robot problems; students must make sense of the problems to inform their solutions
MP2 Reason abstractly and quantitatively	Programming requires students to reason about physical quantities in the world to plan a solution, then calculate or estimate them for the robot
MP4 Model with mathematics	Many processes, including the process of programming itself, must be systematically modeled on both explicit and implicit levels
MP6 Attend to precision	Robots require precise (and accurate) input, or their output action will be correspondingly sloppy
MP7 Look for and make use of structure	Understanding the structure of the physical environment, the interrelated components of robot hardware and software, and commands within a program are vital to successful solutions
MP8 Look for and express regularity in repeated reasoning	Any programmed solution to a class of problems relies on the programmer recognizing and exploiting important patterns in the problem structure. There is also an emphasis throughout the module on recognizing common programmatic patterns, as well as patterns within a solution that invite the use of Loops.

Common Core Mathematics Content

Standard (CCSS.Math.Content)	Introduction to Programming the EV3
6.RP.A.1 Understand the concept of a ratio and use ratio language to describe a ratio relationship between two quantities	Students use ratio language to describe and make use of the relationship between quantities such as Wheel Rotations and Distance Traveled
6.RP.A.2 Understand the concept of a unit rate a/b associated with a ratio $a:b$ with $b \neq 0$, and use rate language in the context of a ratio relationship	The relationship between Wheel Rotations and Distance Traveled is a rate, customarily understood through a unit rate such as "# cm per rotation".
6.RP.A.3 Use ratio and rate reasoning to solve real-world and mathematical problems	Students are required to apply ratios and rates when they build their prototype examples of their real world robots.
7.RP.A.3 Use proportional relationships to solve multistep ratio and percent problems.	Comparisons between rate-derived quantities are common during robot navigation tasks.

Common Core English Language Arts





Introduction to Programming LEGO® MINDSTORMS® EV3

What Standards does the Introduction to Programming EV3 Curriculum address? (continued)

Standard (CCSS.ELA-Literacy)	Introduction to Programming the EV3
WHST.6-8.1 Write arguments focused on discipline-specific content. [See also: WHST.6-8.1.a to WHST.6-8.1.e]	Reflection Questions ask students to analyze, evaluate, and synthesize arguments in response to robotics and programming problems
WHST.6-8.4 Produce clear and coherent writing in which the development, organization, and style are appropriate to task, purpose, and audience.	Reflection Question tasks include composing technical critiques, technical recommendations, and creative synthesis.

Next Generation Science Standards (NGSS)

Standard	Introduction to Programming the EV3
MS-ETS1-2. Evaluate competing design solutions using a systematic process to determine how well they meet the criteria and constraints of the problem.	Solving challenges requires students to create and evaluate both hardware and software designs according to scenario scoring criteria. Some Reflection Questions require students to make recommendations between competing alternatives based on criteria that they define.
MS-ETS1-4. Develop a model to generate data for iterative testing and modification of a proposed object, tool, or process such that an optimal design can be achieved.	When solving more difficult and complex challenges, students are guided toward iterative testing and refinement processes. Students must optimize program parameters and design.
HS-ETS1-2. Design a solution to a complex real-world problem by breaking it down into smaller, more manageable problems that can be solved through engineering.	Problem Solving methodology for challenges directs students to break down large problems into smaller solvable ones, and build solutions up accordingly; challenges give students opportunities to practice, each of which is based on a real-world robot
HS-ETS1-3. Evaluate a solution to a complex real-world problem based on prioritized criteria and trade-offs that account for a range of constraints, including cost, safety, reliability, and aesthetics as well as possible social, cultural, and environmental impacts.	Some Reflection Questions require students to make recommendations about real-world policies (e.g. requiring sensors on automobiles) based on the impact of that decision





Introduction to Programming LEGO® MINDSTORMS® EV3

What Standards does the Introduction to Programming EV3 Curriculum address? (continued)

Computer Science Principles Framework (CSP)

Learning Objective	Introduction to Programming the EV3
1.1.1 Use computing tools and techniques to create artifacts. [P2]	Challenge activities result in the creation of a (simple) algorithmic solution and an accompanying program that implements it.
1.1.2 Collaborate in the creation of computational artifacts. [P6]	Students work in teams to accomplish tasks.
1.1.3 Analyze computational artifacts. [P4]	Students perform debugging on their own code, as well as analyze and evaluate others' code and suggested code in Reflection Questions.
1.3.1 Use programming as a creative tool. [P2]	Students use programming to solve model challenges based on challenges real robots face.
2.2.1 Develop an abstraction. [P2]	Robots gather information about the world through sensors, which turn physical qualities of the world into digital abstractions. Students must understand and work with this data to develop then implement their solution algorithms.
2.3.1 Use models and simulations to raise and answer questions. [P3]	Students construct and use a "program flow" model of programming itself to understand how the robot uses data to make decisions and control the flow of its own commands.
4.1.1 Develop an algorithm designed to be implemented to run on a computer. [P2]	Students develop solution algorithms to each challenge and mini-challenge problem before implementing them as code. Reflection Questions also ask students to evaluate algorithms expressed as pseudocode.
4.2.1 Express an algorithm in a language. [P5]	Students develop code to robotics challenges in the EV3 Programming Language.
5.1.1 Explain how programs implement algorithms. [P3]	Students must communicate solution ideas within groups and as part of class discussion, as well as in Reflection Questions.
5.3.1 Evaluate a program for correctness. [P4]	Students test and debug their own code, and evaluate others' in the Reflection Questions.
5.3.2 Develop a correct program. [P2]	Programmed solutions to challenges must work.
5.3.3 Collaborate to solve a problem using programming. [P6]	Students develop solutions in teams.
5.4.1 Employ appropriate mathematical and logical concepts in programming. [P1]	Relationships such as "distance per wheel rotation" are important to making solutions work.
7.4.1 Connect computing within economic, social, and cultural contexts. [P1]	Reflection Questions ask students to make evaluative recommendations based on the impacts of robotic solutions in context.





Introduction to Programming LEGO® MINDSTORMS® EV3

What is the best setup for student workstations?

Ideally, each pair of students will work together at one computer, with one EV3 robot.

Set up each workstation with:

- **LEGO® MINDSTORMS® Education EV3 Programming Software** installed from its DVD
 - Education version required*
- Access to the **Introduction to Programming LEGO® MINDSTORMS® EV3** curriculum software
 - This can be installed locally or on a local network server via DVD
 - This may also be accessed remotely via internet, if your school's network infrastructure and policies allow
- Two pairs of **headphones** with **headphone splitters**
 - One pair for each student
 - Avoid using speakers, as multiple workstations in the same classroom will generate too much overlapping noise
- One **45544 LEGO® MINDSTORMS® Education Set**

What are the System Requirements for the Introduction to Programming EV3 Curriculum?

LEGO® MINDSTORMS® Education EV3 Programming Software: See packaging

Introduction to Programming EV3 Curriculum

- HTML5-compatible browser (Firefox, Chrome, Internet Explorer 10+)
- Tablets (iPad, Android, Windows) with HTML5 browsers should work as well, if accessing the curriculum from the Internet

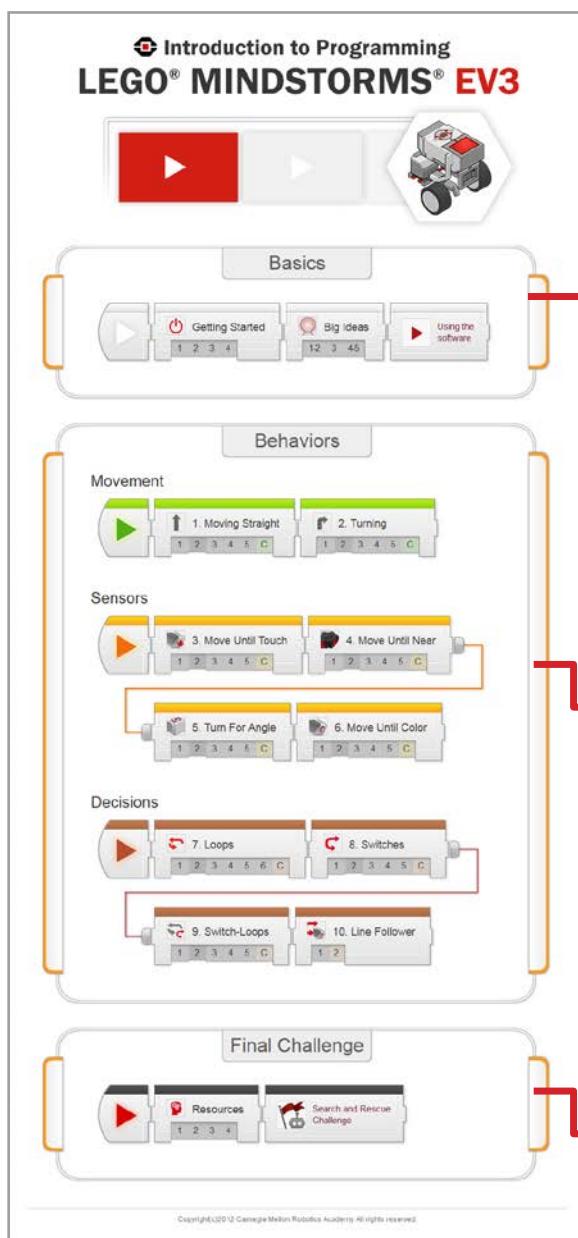
* Retail versions of the EV3 set do not include the Gyro Sensor or software support for it by default





Introduction to Programming LEGO® MINDSTORMS® EV3

*What is the general layout of the
Introduction to Programming EV3 Curriculum?*



Introduction to Programming LEGO® MINDSTORMS® EV3

This is the main menu. Click any section to open the first step, or click a page number to go directly to the page.

Basics Unit

Getting Started: Set up the robot and learn about its basic operation and maintenance

Big Ideas: Five big ideas that will be important throughout the course

Using the Software: General usage patterns in the EV3 Programming Software

Behaviors Unit

Movement: Use sequential commands to make the robot move and turn

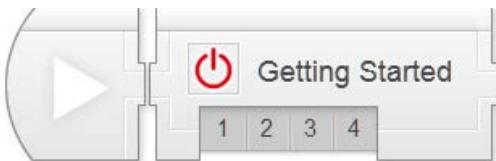
Sensors: Use Sensors to stop the robot in different situations

Decisions: Use Loops and Switches to control the program with smarter decisions

Final Challenge Unit

Combine the techniques of earlier units to tackle a more complex challenge





Basics - Getting Started

Basics > Getting Started Chapter

The Getting Started portion is designed to get a new EV3 user up and running as quickly as possible. Instructors should follow along with all steps, but the first two pages in the chapter could be considered optional for students.

Key Concepts: EV3 operation and maintenance, battery requirements, firmware

► **Getting Started 1: Batteries**

Walks through the options and procedures for powering the EV3

► **Getting Started 2: Firmware**

Introduces the concept of firmware and walks through the process of updating the EV3 to the latest version

► **Getting Started 3: Ports**

Identifies and describes the functions of the various ports on the EV3

► **Getting Started 4: Menus**

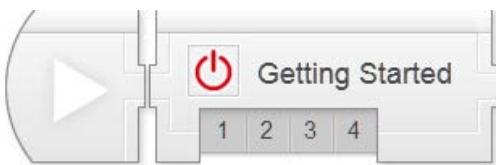
Walks through and explains the main areas and functions available through the EV3's on-screen menus

Hints:

- Go through Battery and Firmware installation prior to using the robots with students. You may want students to view these videos as well for familiarity, should issues arise.
- The Ports (3rd) and especially Menus (4th) videos are very helpful for students.

Teacher's Edition Note: This chapter's notes will point out common structural elements such as Check Your Understanding Questions. In later sections, common notes will be omitted, as they are the same throughout the product.





Basics - Getting Started

Getting Started 1 : Batteries

Batteries

Topics Covered

Batteries

Check Your Understanding:

Q 1. Which of the following CANNOT be used for providing power the EV3?

- 6 AA batteries
- 6 AAA batteries
- EV3 Rechargeable DC Battery
- The EV3 accepts any of the sources listed above

Next

Getting Started 1: Batteries

This step walks you through the process of charging and installing the rechargeable battery pack in the EV3.

Batteries Video

Walks you through the process of charging and installing the battery. Follow along with the video as it covers each step!

Check Your Understanding Questions

These questions are designed to quickly check comprehension of the topics covered in the video. Answer them before moving on.

Getting Started 2 : Downloading Firmware

Downloading Firmware

Topics Covered

Download Firmware

Check Your Understanding:

Q 1. What is Firmware?

- A special type of software that allows the EV3 to understand the programs you write
- A special type of software used by law firms
- The parts of the robot that are not flexible
- The physical appearance of the robot

Q 2. What software do you use to download firmware onto the robot?

- The special EV3 Firmware Downloader app for mobile phones
- The Introduction to Programming: LEGO MINDSTORMS EV3 curriculum software
- The EV3 Programming Software
- The firmware can only be updated at the factory

Q 3. Where in the software is the option to download firmware?

- In the Tools Menu, under Firmware Update
- On the main menu, as a button marked "Download Firmware"
- In the EV3's on screen menus, under "Firmware Tools"
- In the lesson pages

Q 4. How do you turn on the EV3?

Getting Started 2: Downloading Firmware

This step explains the idea behind Firmware, and walks you through the process of loading firmware onto the robot.

Firmware Video

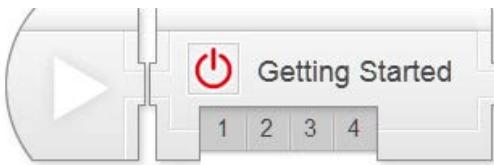
Walks you through the process of loading Firmware onto the robot. Follow along with the video as it covers each step!

Check Your Understanding Questions

These questions are designed to quickly check comprehension of the topics covered in the video. Answer them before moving on.

Bottoms of pages will sometimes be omitted from the Teacher's Guide if they do not contain important notes.

That is the case here – the next page of the guide will go straight to the third Getting Started video.



Basics - Getting Started

A screenshot of the 'Ports' video player. The title 'Getting Started 3 : Ports' is at the top. Below it is a video thumbnail showing a robot with two motors and two sensors. To the right of the video is a 'Topics Covered' section with a single bullet point 'Ports'. At the bottom is a 'Check Your Understanding:' section containing two questions about motor and sensor port identification. Buttons for 'Previous' and 'Next' are at the bottom.

Getting Started 3: Ports

This step provides an overview of the Ports on the EV3 brick.

Ports Video

Identifies and explains the function of each port on the EV3 brick.

Check Your Understanding Questions

These questions are designed to quickly check comprehension of the topics covered in the video. Answer them before moving on.

A screenshot of the 'Menus' video player. The title 'Getting Started : Menus' is at the top. Below it is a video thumbnail showing a robot. To the right of the video is a 'Topics Covered' section with a single bullet point 'Menus'. At the bottom is a 'Check Your Understanding:' section containing two questions about turning the EV3 on and off. Buttons for 'Previous' and 'Next' are at the bottom.

Getting Started 4: Menus

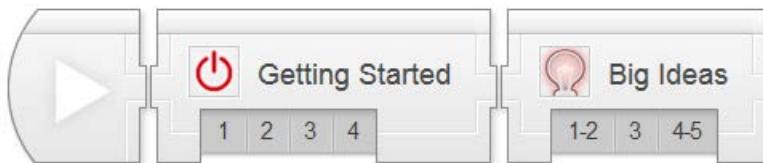
This step explains the general interface conventions and tours the main areas of the EV3 on-brick menu interface.

Menus Video

Shows you how to use the EV3 on-screen menus to control robot settings, run programs, and more.

Check Your Understanding Questions

These questions are designed to quickly check comprehension of the topics covered in the video. Answer them before moving on.



Basics - Big Ideas

Big Ideas 1 :

Topics Covered

- Idea #1: Programming is Precise
- Idea #2: Sensors, Programs, Actions

Check Your Understanding:

Q. 1. What does the video mean by, "Programming is Precise"?

- The robot will always move to the precise location you want
- You need to tell the robot precisely what to do, or it might not be what you wanted
- If you do not type the commands exactly as shown, the program will crash
- The robot will do precisely what you want, even if you give it the wrong command

Q. 2. What do Sensors do for a robot?

- Provide power to onboard electronics
- Create a physical effect like turning a wheel
- Processes data and makes decisions
- Give it data on its position and surroundings

Q. 3. What does a robot's Program do?

- Provide power to onboard electronics
- Create a physical effect like turning a wheel
- Process data and make decisions
- Give it data on its position and surroundings

Big Ideas 1 & 2

Big Ideas 1 & 2 Video

This video introduces two concepts:

Big Idea #1: Programming is Precise

If you want a robot to do something, you need to communicate that idea with mathematical and logical precision, or it won't quite be what you intended

Big Idea #2: Sensors, Programs, and Actions

Data from sensors gives a robot information about its environment. A program uses that data to make decisions, and the robot Acts on those decisions. Data underlies the core of the entire process

Check Your Understanding Questions

These questions are designed to quickly check comprehension of the topics covered in the video. Answer them before moving on.

Big Ideas 2 :

Topics Covered

- Idea #3: Make Sense of Systems

Check Your Understanding:

Q. 1. What kind of Model is the video talking about?

- A "mental model" – a systematic understanding of the way something works
- A design for a LEGO robot "model"
- A tiny "model" version of the real robot
- A "model" who appears in professional photography

Q. 2. What is one advantage of a "model"-based understanding over just memorizing facts?

- You can learn a "model" faster than memorizing every single fact
- You can reason about new situations, even ones you haven't seen before
- Models make things make sense
- All of the above

Big Idea 3

Big Idea 3 Video

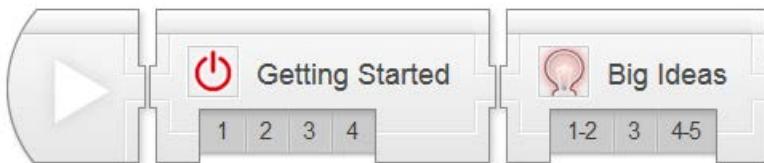
This video introduces:

Big Idea #3: Make Sense of Systems

To understand the way something works, construct a mental "model" of it in your head that captures the important features and rules of the system. This helps you make sense of it, and also gives you a tool to "play out" (similar) new scenarios in your head to predict what would happen

Check Your Understanding Questions

These questions are designed to quickly check comprehension of the topics covered in the video. Answer them before moving on.



Basics - Big Ideas

Big Ideas 3 :

Topics Covered

- Idea #4: Break Down Problems and Build Up Solutions
- Idea #5: Computational Thinking Applies Everywhere**

Check Your Understanding:

- What does the video mean by, "Break down problems"?
 - Split up a big problem into smaller ones that are easier to solve
 - Solve the problem all at once
 - Give up because the big problem is too hard
 - Technical difficulties with the robot
- What does the video mean by, "Build up Solutions"?
 - Build a fully working solution by expanding upon partial solutions
 - The best programs solve every problem at the same time
 - Build your robot starting with the wheels and working your way upward
 - Accumulate a lot of chemical compounds
- Which of the following is NOT true about Computational Thinking?
 - It is applicable outside of computer programming
 - It includes the idea of using models to understand systems
 - It is used in robotics
 - It refers only to the process of writing code for a program

Previous **Complete!**

Big Ideas 4 & 5

Big Ideas 4 & 5 Video

This video introduces two concepts:

Big Idea #4: Break Down Problems and Build Up Solutions

To solve a difficult problem, try breaking it down into smaller problems. Then, solve the smaller problems, building up toward a solution to the big problem.

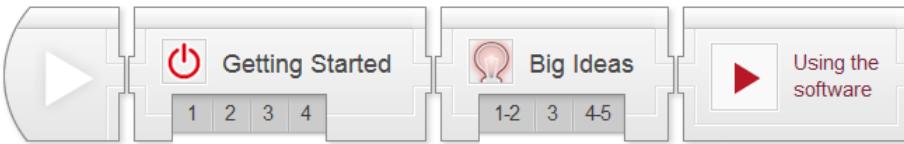
Big Idea #5: Computational Thinking Applies Everywhere

These skills – mathematical and logical clarity, using data, systems thinking with mental models, and problem solving – are not just for robotics. They are key to solving many problems in the world.

Check Your Understanding Questions

These questions are designed to quickly check comprehension of the topics covered in the video. Answer them before moving on.

Basics: Using the EV3 Software



Getting Started : How to use the Software

Topics Covered

- Lego MINDSTORMS Education EV3 Software

Using the EV3 Programming Software

Check Your Understanding:

Q 1. Where can robot building instructions be found?

- The EV3 set comes pre-assembled in the only supported configuration
- In the EV3's firmware menus
- In the printed manual that came with the EV3 Set
- In the Robot Educator section of the EV3 Education Programming Software

Q 2. Most project files will contain:

- Over 10,000 different programs in 10 different languages
- Other Project files
- A single program, called Program
- Robot files3

Q 3. Which of the following connection types allows you to download programs onto the robot?

- USB cable
- Ethernet cable
- IR connection
- Power cable

Q 4. On the picture below, click the button that loads the program onto the robot but does not immediately run it.

Show Answer

Q 5. Which of the following will help you navigate around a large program?

- Scroll arrows on the sides of the screen
- Zoom controls in the top-right corner of the window
- Arrow keys on the keyboard
- All of the above

Q 6. How do you add a comment to your program?

- Start typing on the keyboard and a comment will automatically appear
- Click the comment tool to create a comment box, then type in the box to add a comment
- Right-click a box and select Add Comment
- The EV3 software does not support adding comments

Complete

Copyright © 2012 Carnegie Mellon Robotics Academy. All rights reserved.

Using the Software

Using the EV3 Software Video

This video provides introductory tips and trick of how to use the EV3 software

- ▶ How to Open the Software
- ▶ Finding Building Instructions
- ▶ Beginning Programming
- ▶ How to Link Blocks
- ▶ Mode Settings
- ▶ Configuring Blocks
- ▶ Saving
- ▶ How to Run the Program
- ▶ Navigating to the Program on the EV3
- ▶ Scrolling on the Software Interface
- ▶ Commenting Your Code

Check Your Understanding Questions

These questions are designed to quickly check comprehension of the topics covered in the video. Answer them before moving on.



Introduction to the Moving Straight Unit

Movement > Moving Straight Chapter

In Moving Straight, students program the robot to move forward, then explore variations such as moving for different distances or at different speeds.

Key Concepts: Writing and running programs, Move Steering Block, Rotations and Distance, Sequential commands

► **Moving Straight 1: Introduction to Sensabot**

Introduces the real-world robot (Sensabot), and the challenge modeled after it (Sensabot Challenge)

► **Moving Straight 2: Robot Config**

Contains building and setup instructions for the rest of the chapter

► **Moving Straight 3: Steering Forward**

Introduces the Move Steering Block and programming the forward movement

► **Moving Straight 4: Arm Control**

Introduces the Medium Motor Block and programming the arm to move

► **Moving Straight 5: Review**

Explains sample solutions to mini-challenges from this chapter

► **The Moving Straight Challenge**

Requires students to make the robot move to 3 marked lines by controlling the distance of each movement

Hints:

- Remember that each chapter is based around the real-world robot theme and challenges. Use these contextual surroundings to help ground discussions and decision-making processes (e.g. “Do you think guessing to find the correct distance would be appropriate, since Sensabot needs to perform this task reliably in the real world?”). [NGSS: MS-ETS1, CCSS.Math.Practice.MP1]
- The Distance a robot moves is (Wheel Circumference * Wheel Rotations). This is because the turning of the robot’s wheels are what propel it along the ground. [CCSS.Math.Practice.MP4, CCSS.Math.Content.7.RP.A.2]

Teacher’s Edition Note: This chapter’s notes will point out common structural elements such as Check Your Understanding Questions. In later sections, common notes will be omitted, as they are the same throughout the product.



Movement - Straight

Moving Straight 1: Introduction

This module introduces the Sensabot real-world robot that the chapter Challenge is based on.

Sensabot Video

Introduces the real-world robot and summarizes the chapter Challenge. Students should watch this video before beginning work on the chapter.

Check Your Understanding Questions

These questions are designed to quickly check comprehension of the topics covered in the video. Students should answer them before moving on.

These generally correspond to Remembering and Understanding tasks in Bloom's Taxonomy.

Higher-level enrichment and assessment will be handled by Try It! exploration items and Mini-Challenge activities in later steps.



Movement - Straight

Moving Straight 2 : Robot Config

Robot base Configuration
For this chapter, EV3 driving base with the medium motor attachment is required to complete sections of this chapter as well for the final challenge.

EV3 DRIVING BASE WITH ARM*
*ACTUAL DESIGN MAY VARY

EDUCATION MODEL WITH ARM
PLEASE REFER TO THE EDUCATION INSTRUCTIONS CONTAINED IN THE EV3 SOFTWARE

Robot Educator Building Instructions → Driving Base 1/18 + **Robot Educator** Building Instructions → Medium Motor - Driving ... 2/16

Where can I find the instructions?

Moving Straight 3 : Steering Forward

Topics Covered

- Start the EV3 Programming Software
- Move Steering Block (Forward)
- Download and Run a program

Steering Forward

Check Your Understanding:

Q 1. What does the robot do when the program is run?
 Move forward until its wheels have turned 1 rotation
 Move forward until its wheels have turned 3 rotations
 Move backward
 Turn to the left

Q 2. How do you create a new program in the EV3 Programming Software?
 Click the + tab near the top of the screen
 Press the New Program button on the home screen
 Enter a name and press the Go >>> button
 Click the Quick Start button on the menu

Q 3. How do you run a program that has been downloaded to the EV3?
 File > Run on the EV3 brick
 My Files > Software Files > Run on the EV3 brick
 "File Navigation" tab > Project name > Program on the EV3 brick

Moving Straight 2: Robot Configuration

The Building Instructions are located inside the LEGO Programming Software. Follow the instructions on this page to view them.

As of EV3 Software version 1.0.1, the instructions cannot be printed, and must be viewed on-screen.

Moving Straight 3: Steering Forward

This step introduces the Move Steering Block, and walks students through programming a straight forward movement.

Step-by-Step Video

Walks students through the process of starting up the software, writing the Moving Straight program, and running it. Make sure students follow along with the instructions as the video plays (pause as needed).

Check Your Understanding Questions

Quickly check comprehension of the topics covered in the video. Students should answer them before moving on.



Movement - Straight

Mini Challenge

Mini Challenge 1: 50 cm Challenge
Program your robot to travel exactly 50 cm!

Place two pieces of black electrical tape 50 cm apart.
Your robot should travel exactly from one to the other to complete the challenge!

[Click Show hint to add hints here]

Moving Straight 3 (cont'd)

Mini-Challenge: 50 cm challenge

Challenges students to make the robot move a specific distance by adjusting the Rotations setting.

Mini-challenges ask students to perform a task that is very closely related to what they have already done in the video-guided portion.

Hints are provided, and can be revealed (one at a time) by clicking the Show Hint button on the box below the picture.

Mini-Challenges typically align to Applying tasks in Bloom's Taxonomy.

Optional Activities

Try It!

Try It! 1 Brake vs. Coast
The last space on the Move Steering Block is for "Apply Brakes". What is the difference between the "Brake" and "Coast" settings for this option?

What happens?

Optional Activity Cutoff Line

Activities below this line are helpful but not critical to the completion of the Challenge. Use these activities to help with class pacing.

[Optional] Try It!: Brake vs. Coast

Prompts students to try changing the Brake setting on the Move Steering Block to "Coast" and see what happens.

Try It! activities prompt students to explore an additional feature or area of the software.

The What Happens? button will show the result so students can confirm their observations.

Sometimes this exploration is for enrichment, but often it is important to the Challenge. Use the Optional Cutoff to identify when an activity is critical.

Did You Notice?

Did you notice? Projects and Programs

LEGO MINDSTORMS Education EV3 Teacher Edition
File Edit Tools Help
Project X Project2 Tab1 Tab2
Program X Program2 X Experiment X +
Did you notice that there are 2 tab bars in the EV3 Programming Software?
The top bar is for Projects, while the bottom one shows what is inside each project, including Programs.

Project Moving Forward

Program 1
Program 1
Program 1

A single Project can contain several Programs. Sometimes one program will run another program, so they need to be kept together. A project can also contain Experiments that use the EV3's sensors to gather data.

[Optional] Did You Know?: Projects and Programs

Explains the relationship between Projects and Programs in the EV3 Programming Software.

Did You Know? activities provide additional background information on various topics.

Q Which of the following is correct about the projects and the program inside it?

A project can contain multiple programs
 A program can contain multiple projects



Movement - Straight

Did you notice? Rename Program

You can rename the "Program" inside your Project by **double-clicking** the word "Program" on the tab, and typing a new name!

Now your program will have a name inside the Project, instead of just being called "Program".

Did you notice? EV3 Menus



Did you notice the four main areas at the top of the EV3's on-screen menu?

- **Run Recent:** Lists the most recent programs you have to run. Not that if they are all named "Program", they might be hard to tell apart!
- **File Navigation:** Browse through all the Projects you have loaded on the EV3. Select and run Programs from inside each Project! Folders starting with "BRK" contain Brick Apps programs (more on that later).
- **Brick Apps:** This mode contains utilities that let you view and set motor and sensor values for troubleshooting purposes. The Brick Program app lets you write simple programs or log sensor values directly on the brick.
- **Settings:** Adjust the speaker volume, auto-sleep timer, networking, and other system settings.

Did you notice? Auto-detecting Ports



Did you notice that the EV3 automatically detected the Ports that your motors were plugged into?

The EV3 features a technology called "AutoID" that allows it to automatically detect, identify, and configure any EV3 hardware plugged into it! However, it can only tell what type of device is plugged in. It cannot detect certain other information like what size your wheels are, or which motor is on the left vs. the right side of the robot.

[Previous](#)

[Next](#)

Copyright©2012 Carnegie Mellon Robotics Academy. All rights reserved.

Moving Straight 3 (cont'd)

[Optional] Did You Know?: Rename Program

Programs can be renamed within Projects, although this is typically only done when a Project contains multiple programs.

[Optional] Did You Know?: EV3 Menus

A brief overview of useful functions in the EV3 on-screen menu system. See the Basics unit or the EV3 manual for more in-depth information.

[Optional] Did You Know?: Auto-Detecting Ports

The EV3 has the built-in capability to locate and identify any motor or sensor plugged into it. This means you don't have to worry about port numbers, as long as there is only one of a given device type attached (e.g. only one Touch Sensor).

However, since there are two motors on the robot, you do need to pay attention to which one is the left and which is the right.



Movement - Straight

Moving Straight 4 : Arm Control

Topics Covered

- Medium Motor Block (Arm)

Check Your Understanding:

- What does the Medium Motor block?
 - Controls both of the large motors.
 - Controls the medium motor like the one on the arm.
 - Controls the ultrasonic sensor.
 - Controls all the motors.
- What happens when you use a negative power level?
 - The motor runs backwards.
 - The robot moves faster.
 - The motor does nothing.
 - The program crashes.
- What happens when you put more than one block in a program?
 - The program runs backwards.
 - The program runs in sequence.
 - The program doesn't run at all.
 - The program runs the largest motor block first.

Try It!

Negative Power

The Move Steering command also accepts negative power values to move backwards. Try entering -50 in the power blank on a Move Steering block.

Mini Challenge

Mini Challenge 1: Cargo Retrieval

Program your robot to:

- Raise its arm
- Move 50cm to the box
- Drop the arm down
- Back up to robot's starting position with the box

Place two pieces of black electrical tape 50 cm apart. Your robot should travel to the block and bring it back to the starting line.

Moving Straight 4: Arm Control

This step introduces the concept of sequential commands. The Medium Motor Block, which controls the single arm motor is added to the program..

Step-by-Step Video

Walks students through adding a second and third command to the program, using the Medium Motor Block to control the robot's arm. Students should follow along with the video.

Check Your Understanding Questions

These questions are designed to quickly check comprehension of the topics covered in the video. Students should answer them before moving on.

Try It!: Negative Power

Negative power levels make the robot move backward.

Mini-Challenge: Cargo Retrieval

Challenges students to build on their previous programs by making the robot move forward, then lower its arm, then move back to its starting location.



Movement - Straight

Optional Activities

Did you notice?
Getting the Program Stuck


What happens if a block cannot complete its action?


Try running your program with its arm already in the "up" position. Watch carefully so you can answer the following questions.

Q. 4. Does the program ever end?
 The program keeps running
 The program ends immediately

Q. 5. Does the second medium motor block ever get to run?
 The second motor block never runs
 The second motor block is skipped over

Q. 6. What happens if a block cannot complete its action?
 The program will immediately skip to the next block
 The program will try for a while, then display an error and quit
 The program will try for a while, then skip the "stuck" block and move on to the remaining commands
 The program will get "stuck" trying to complete the action, and later blocks will never be run

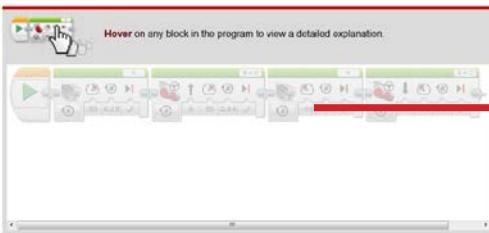
Previous **Next**

Copyright©2012 Carnegie Mellon Robotics Academy. All rights reserved.

Moving Straight 5 : Moving Forward Review

Program Review: Cargo Retrieval

- The program shown below is a sample solution to the Cargo Retrieval Mini-Challenge. The program makes the robot move forward and retrieve a cargo box 50 cm away, then return home with it.



Previous **Next**

Copyright©2012 Carnegie Mellon Robotics Academy. All rights reserved.

Moving Straight 4 (cont'd)

Did You Know?: Getting the Program Stuck

Explains one of the common problems with sequentially-executed programs: if any of the commands cannot complete (e.g. if the arm hits the ground and cannot lower any further), the entire program will be “stuck” and unable to progress.

Check Your Understanding Questions

These questions are designed to quickly check comprehension of the topics covered in the video. Students should answer them before moving on.

Moving Forward 5: Review

Review steps come at the end of the guided portion of a unit, just before the main Challenge. Sample solutions to the Mini-Challenges are shown here in picture format.

Placing the mouse cursor over any block in the program will show a detailed explanation identifying the block, describing the literal command it issues, and explaining what action it performs in context.



Movement - Straight

Moving Straight Challenge

This step lays out the details for the Sensabot Chapter Challenge. Students should work in their teams to complete the challenge objectives.

Challenge Video

Describes the challenge in video format. The robot must move to three marked locations on a game board, and raise and lower its arm at each location to represent taking a sensor reading.

Challenge Diagram

A non-technical summary of the Sensabot Chapter Challenge.

Challenge PDF

A link to the official rules and gameboard layout for the Challenge in PDF format. Detailed measurements for the board layout, as well as instructions for setting it up, rule details, and hints for solving the challenge can be found in the attached printable document.

Sensabot Challenge PDF

Detailed measurements for the board layout, as well as instructions for setting it up, rule details, and hints for solving the challenge can be found in this printable document.

Moving Straight Challenge

This step lays out the details for the Sensabot Chapter Challenge. Students should work in their teams to complete the challenge objectives.

Challenge Video

Describes the challenge in video format. The robot must move to three marked locations on a game board, and raise and lower its arm at each location to represent taking a sensor reading.

Challenge Diagram

A non-technical summary of the Sensabot Chapter Challenge.

Challenge PDF

A link to the official rules and gameboard layout for the Challenge in PDF format. Detailed measurements for the board layout, as well as instructions for setting it up, rule details, and hints for solving the challenge can be found in the attached printable document.

Sensabot Challenge PDF

Detailed measurements for the board layout, as well as instructions for setting it up, rule details, and hints for solving the challenge can be found in this printable document.



Introduction to the Turning Unit

Movement > Turning Chapter

In Turning, students program the robot to turn in place, then explore variations such as turning in the opposite direction, and turning in a wide arc.

Key Concepts: Turning, Types of Turns, Steering setting

- ▶ **Turning 1: Introduction to Autonomous Tractor**
Introduces the real-world robot (Autonomous Tractor), and the challenge modeled after it (Orchard Challenge)
- ▶ **Turning 2: Robot Configuration**
Contains building and setup instructions for the rest of the chapter
- ▶ **Turning 3: Turning in Place**
Uses the Steering slider on the Move Steering Block to create turns
- ▶ **Turning 4: Other Turns**
Uses other settings on the Steering slider to create wide turns
- ▶ **Turning 5: Review**
Explains sample solutions to mini-challenges from this chapter
- ▶ **The Orchard Challenge**
Requires students to program the robot to navigate rows of trees in an orchard

Hints:

- ▶ The “angle” turned by a robot is generally in reference to its heading. A 90 degree turn means that the robot’s heading has changed by 90 degrees.
- ▶ When turning, a robot’s wheels travel along a curved path. The shape and angle of the turn is determined by how far each wheel travels along that path. [CCSS.Math.Practice.MP4, CCSS.Math.Content.7.RP.A.2]

Teacher’s Edition Note: Notes on the purpose of common structural elements such as Check Your Understanding Questions are omitted from this section onward. To review these general notes, please see the Movement > Moving Straight chapter.



Movement - Turning

Turning 1 : Introduction with Crop Tractor

Autonomous Tractor
Orchard Challenge

Topics Covered

- Autonomous Tractor
- Challenge overview

Check Your Understanding:

Q 1 Why is it important to be able to drive through an orchard?

- To perform specialized tasks to different types of crops
- GPS is required while navigating around the orchard
- To perform tasks like inspection and spraying which cannot be done as effectively through other means
- All of the above

Q 2 What is the advantage of the Autonomous Tractor over a human driver?

- Reduces the need for humans to perform the repetitive task of driving through the orchard over and over
- Reduces exposing human to hazard areas while performing inspections
- Autonomous Tractor can travel through an area where a human driver may get lost
- There is no big advantage

Q 3 In addition to basic turning, what additional, new knowledge will help you complete this challenge?

- How a robot moves straight
- How a robot move back and forth
- How a robot turns, and different types of turns
- All of the above

Turning 2 : Robot Config

Robot base Configuration

For this chapter, EV3 driving base is required to complete sections of this chapter as well for the final challenge.

EV3 DRIVING BASE

EDUCATION MODEL
PLEASE REFER TO THE BUILDING INSTRUCTION CONTAINED IN THE EV3 SOFTWARE

Robot Educator Building Instructions Driving Base 1/16

Where can I find the instructions?

Turning 1: Introduction

Introduces the Autonomous Tractor and the Orchard Challenge.

Autonomous Tractor Video

Introduces the real-world robot and summarizes the chapter Challenge. Students should watch this video before beginning work on the chapter.

Check Your Understanding Questions

These questions are designed to quickly check comprehension of the topics covered in the video. Students should answer them before moving on.

Turning 2: Robot Configuration

Building Instructions for the robot are the same as Moving Straight.



Movement - Turning

Turning 3 : Turning in Place

Topics Covered
Move Steering Block (Turning)

Turning In Place

Check Your Understanding:

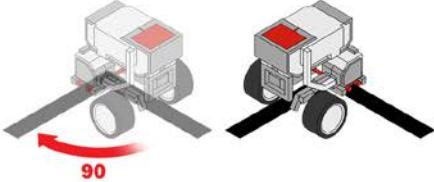
Q 1. What does the robot do when the TurnRight program is run?
 Move straight forward
 Spin to the robot's right without moving forward at all
 Spin to the robot's left without moving forward at all
 Spin for 360 degrees

Q 2. TRUE or FALSE: With "Rotations" on the Move Steering Block to 1, the whole robot rotates 1 time.
 TRUE: the robot will turn around 1 time
 FALSE: the wheels will turn 1 time, not the body

Mini Challenge

Mini Challenge 1: 90 Degree Turn
 Program your robot to turn exactly 90 degrees to its right!

Place two pieces of tape so they form 90 degree angle.
 Place your robot that it faces along one piece of tape, then program it so that it turns to face directly along the next piece of tape.



(Click Show hint to add hints here) Show hint

Optional Activities

Try It!

Try It! 1 | Direction of Turn
 Moving the Steering slider all the way to the right makes the robot turn to the right, in place. What happens if you move it all the way to the left?



What happens?

Turning 3: Turning in Place

This step introduces the Steering slider to make the robot turn to the side. For now, only the center and ends of the slider are used. Intermediate settings produce “wide” turns and are explored in the next step.

Mini-Challenge: 90 Degree Turn

Challenges students to make the robot turn a specific amount (90 degree change in the direction the robot is facing).

This activity is directly analogous to the “50 cm mini-challenge” in the previous chapter, and pointing out the parallels in the two problems is appropriate.

The ratio of Wheel Rotations to Body Turn on the REM-EV3 driving base model is approximately 2:1; that is, it takes 2 wheel rotations to turn the robot around exactly 1 time. Turning the robot 90 degrees, then, would take 0.5 wheel rotations (180 degrees).

[Optional] Try It!: Direction of Turn

Prompts students to try moving the Steering slider to the other side. This produces a turn in the opposite direction.

The direction of turn always matches what you would get if you turned a steering wheel in the same direction (“left” or “right”).



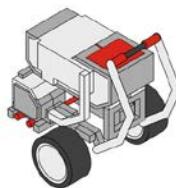
Movement - Turning

Did You Notice?

Did you notice?
Wheel Pointers



The white pointers on the EV3's tires help you to see how much the wheels are rotating.



Run your TurnRight program again, and watch the pointer on the robot's right wheel

Q. 3. How much did the robot's wheel turn during this movement?

- 1 rotation
- 1 degree
- Enough to make the robot spin completely around one time
- One lap around the table

Q. 4. What does the "1 rotation" refer to in the Move Steering Block's controls?

- 1 full rotation of the robot's body during a turn
- 1 rotation of the robot's wheels
- 1 time that the robot is picked up and turned around
- 1 rotation of the Earth and its axis

Previous

Next



Turning 4 : Other Turns



Topics Covered

- Move Tank Block
- Types of Turns

Check Your Understanding:

Q. 1. In the movement you programmed, the left motor was told to move forward at 50% power, and the right motor was told to...

- Move forward at 50% power
- Move backwards at 50%
- Stay in place
- Spin freely

Q. 2. What kind of turn did the robot produce with one motor running and one motor stopped?

- Goes straight

Turning 3 (continued)

[Optional] Did you notice?: Wheel Pointers

Calls attention to a design feature of the robot: pointers on the robot's wheels help you see which direction each wheel is turning, and how much.

The amount a wheel turns (e.g. 360 degrees of wheel rotation) is proportional – but not identical – to the amount the robot's body will turn.

Turning 4: Other Turns

This step introduces off-center “wide” turns using the Move Tank Block, which controls the robot’s two wheel motor powers separately.

Any time the robot’s wheels move at different speeds, the robot’s path will curve. The specific shape of the curve is determined by the combination of powers used.



Movement - Turning

Mini Challenge

Mini Challenge 1: Dizzy Drill
Program your robot to run out to an obstacle, go around it, then come back.

Create a starting box for the robot and use a cylindrical object as the obstacle in the middle.

PROGRAM THE ROBOT TO MOVE OUT AND MANEUVER COMPLETELY AROUND AN OBJECT THEN RETURN TO ITS STARTING POSITION.

[Click Show hint to add hints here] **Show hint**

Optional Activities

Try It!

Different Motions
You can create many different type of motion by combining different motor speeds. Try each of the following to see what you get!

Try it! 1 **Different Motions**

What happens?

Previous

Next

Copyright©2012 Carnegie Mellon Robotics Academy. All rights reserved.

Turning 4 (continued)

Mini-Challenge: Dizzy Drill

Challenges students to make the robot move around an obstacle using a combination of straight moves and turns (wide or in-place).

[Optional] Try It!: Different Motions

Prompts students to try different combinations of motor powers. This produces differently curved turns.

Helper Activity: Role-play the robot

One student acts as the robot, while another student issues left-foot and right-foot commands with different power levels.

For instance, “left foot 50; right foot 0” means taking a small step forward with the left foot and holding the right foot in place. Straightening out your body from this stance will make you turn slightly to the right, pivoting around your right foot just as the robot does with a Move Tank Block set to left 50, right 0.



Movement - Turning

Turning 5 : Turning Review

Program Review: Turn in Place 90 degrees

- The program shown below is a sample solution to the 90 degree turn Mini-Challenge. The robot turns 90 degrees to its right, using a Move Steering Block with steering set to +100 (turn to the right in place).
- Note:** This program assumes you are using the **default EV3-RE design**, and is only approximate – your robot may require a slightly different number of rotations.

Review the program below.

Move Steering Block
Makes a point turn to the right...
for 0.5 rotations (which causes about 90 degrees of body turn) at 50% power
Turns the robot 90 degrees to the right

Program Review: Dizzy Drill (Square Turns)

- The program shown below is a sample solution to the Dizzy Drills Mini-Challenge. This solution uses multiple 90-degree turns to navigate around the obstacle.

Hover on any block in the program to view a detailed explanation.

Program Review: Dizzy Drill (Wide Turns)

- The program shown below is a sample solution to the Dizzy Drills Mini-Challenge. This solution uses a single "wide" U-shaped turn to navigate around the obstacle.

Hover on any block in the program to view a detailed explanation.

Previous Next

Copyright ©2012 Carnegie Mellon Robotics Academy. All rights reserved.

Turning 5: Turning Review

Sample solutions to the mini-challenges can be found on this page.

Dizzy Drills has two sample solutions provided: one for a strategy that uses two right-angle turns to go around the back of the obstacle, and one that uses a single U-shaped turn to swing around the obstacle.



Movement - Turning

Turning 6 : Orchard Challenge

Challenge Review

Orchard Challenge

In this challenge, you will program your EV3 robot to move from its starting area through three rows of fruit trees. You may choose your own path through the orchard, but the robot must pass along both sides of each row.

The starting area can be re-located to wherever space is available.

Challenge PDF [orchard_challenge.pdf]

Previous **Complete!**

Copyright©2012 Carnegie Mellon Robotics Academy. All rights reserved.

Orchard Challenge

This step lays out the details for the Orchard Challenge. Students should work in their teams to complete the challenge objectives.

Teams get to choose where they start their robot for this challenge. It is to their advantage to pick a specific spot and use it every time, rather than moving the start position or aligning the robot sloppily.

The route shown in the picture is just an example, not the required (or necessarily easiest) route.

Retain this Challenge board intact, if possible, as it will be re-visited in the Switch-Loops chapter.

Introduction to Programming
LEGO® MINDSTORM® EV3

CHAPTER 2: Orchard Challenge

In this challenge, you will program the robot to move from its starting area through three rows of fruit trees. You may choose your own path through the orchard, but the robot must pass along both sides of each row during its run.

Rules and Procedures:

- For this challenge, the user can create their starting area wherever on the board.
- Use three strips of electrical tape to mark three "rows" of trees. Exact location are not important, as they should not be moved once the board is finalized.
- Make sure there is enough space between the rows for the robot to pass on both sides of each row without crossing the lines.

RIVER **RIVER**

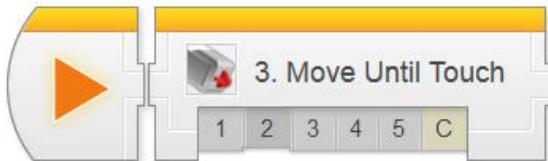
Starting Area **Can Be Placed Anywhere**

Hints:

- Use a meter stick or ruler to measure the distances to each line on the board so you know how far you need to move each time.
- Try finding the number of centimeters your robot travels or number of degrees its body turns in each wheel rotation.
- You can also make the test run, then calculate "how many times as far" you need to move or turn to get the amount of movement you want, compared to a test run.

Orchard Challenge PDF

Detailed measurements for the board layout, as well as instructions for setting it up, rule details, and hints for solving the challenge can be found in this printable document.



Introduction to the Touch Sensor Unit

Sensors > Move Until Touch Chapter

In Move Until Touch, students program a robot to wait until its Touch Sensor is pressed before proceeding with other commands, then combine that behavior with Motors On and Motors Off commands to make the robot stop and go based on Touch Sensor commands.

This chapter is an exception to the pattern of “bookending” every set of activities with real-world robots and challenges. While the chapter is still challenge-based, it is focused on the idea of sensors rather than a specific robot that uses them.

Key Concepts: Wait Block, Touch Sensor, Forward Until Pattern

► **Touch 1: Introduction to Sensors**

Introduces the idea of Sensors, the Touch Sensor, and some of the Touch Sensor-based behaviors in the lesson

► **Touch 2: Robot Configuration**

Contains building and setup instructions for the rest of the chapter

► **Touch 3: Wait for Touch**

Uses the Wait for Touch Block to make a robot wait for a Touch Sensor press before continuing

► **Touch 4: Forward until Touch**

Combines Motor On and Motor Off commands with the Wait for Touch Block to make the robot move forward until the Touch Sensor is pressed

► **Touch 5: Review**

Explains sample solutions to mini-challenges from this chapter

► **The Arm Position Challenge**

Requires students to program a Touch Sensor-based “raise arm” behavior that moves the arm to its “closed” position regardless of where the arm starts.

Hints:

- All of the chapters in the Sensors unit follow the general pattern laid out in this chapter: Introduction, Wait for (Sensor), Forward until (Sensor), Challenge.
- The Touch Sensor does not have a real-world robot because simple touch switches are not commonly found on real-world robots; typically, robots shouldn’t physically collide with their surroundings to sense them.



Touch Sensor

Touch 1 : Introduction

Introduces the idea of Sensors on the robot.

This chapter does not use a real-world robot. Instead, it focuses on the first use of sensors on the EV3 robot.

Topics Covered

- Touch Sensor
- Challenge overview

Check Your Understanding:

Q 1. Why are sensors important to robots?

- They help robots tasks in series
- They give the robot information about its surroundings
- They allow robots to repeat similar tasks
- All of the above

Q 2. What is the advantage of Sensor Control over Sequential Commands?

- The robot can remember hazard areas
- The robot can perform actions a lot faster
- The robot can react to its environment
- There is no big advantage

Next

Copyright©2012 Carnegie Mellon Robotics Academy. All rights reserved.

Touch 2 : Robot Config

The robot is the driving base from the Movement Unit, with a Touch Sensor added.

TOUCH SENSOR ATTACHMENT

TOUCH SENSOR PLACEMENT
PLEASE REFER TO THE BUILDING INSTRUCTIONS CONTAINED IN THE EV3 SOFTWARE

Building Instructions

- Robot Educator → Driving Base 1/18
- + Robot Educator → Touch Sensor - Driving B 3/18

Where can I find the instructions?

Touch 1: Introduction

Introduces the idea of Sensors on the robot.

This chapter does not use a real-world robot. Instead, it focuses on the first use of sensors on the EV3 robot.

Touch 2: Robot Config

The robot is the driving base from the Movement Unit, with a Touch Sensor added.



Touch Sensor

Touch 3 : Wait for Touch

Topics Covered

- Wait Block
- Touch Sensor (Wait for Pressed)

Wait for Touch

Check Your Understanding:

- What does the robot do when the WaitTouch program runs?
 - Runs continuously until the Touch Sensor is pressed in
 - Waits for 1 second, then moves 1 rotation
 - Waits for the Touch Sensor to be pressed in, then moves 1 rotation
 - Runs for 1 rotation
- The program waits BEFORE it moves because...
 - The Wait Block comes first in the program
 - The Wait Block always takes priority over Move Blocks

Try It!

Try It! 1 Already Pressed
What happens if you're already holding down the Touch Sensor's button when you start running the program?

Try It! 2 EV3 Buttons
The 5 buttons on the front of the EV3 (not counting the Cancel button) can be used as Touch Sensors! Try changing the Mode of the Wait Block to "Brick Buttons > Compare > Brick Buttons" and running your program.

Once it's running, press the middle button on the front of the EV3!

Touch 3: Wait for Touch

This step introduces both the Touch Sensor and the Wait For (Sensor) Block.

The Touch Sensor physical mechanism is explained at the bottom of the page.

The Wait For (Sensor) Block “holds up” the program’s flow until the sensor condition is met. When the condition is met, the program continues (with a Move Steering block in this case). This is the simplest way in which the sensor can be used to control a behavior, and is used throughout the Sensors Unit.

Try It!: Already Pressed

Since the Wait for Touch Block literally waits for the “Pressed” state, it can be triggered instantly if the sensor is already pressed in when the program starts.

This is a common source of error with all sensors: if the sensor is already in a “triggered” position, the “wait” is invisibly short. This is often mistaken for the program “ignoring” the sensor.

Try It!: EV3 Buttons

The buttons on the front of the EV3 are essentially touch sensors, and can be used as such.



Touch Sensor

Touch 3 (continued)

Optional Activities

Did You Know?

Did you know?
How the Touch Sensor Works

When the Touch Sensor is pressed, it closes an electrical circuit, allowing current to flow. If the Touch Sensor is released, the circuit is broken and no current flows.

The flow (or lack) of current is detected by the NXT, allowing it to determine if the Touch Sensor is pressed.

Previous Next

Copyright © 2012 Carnegie Mellon Robotics Academy. All rights reserved.

[Optional] Did You Know?: How the Touch Sensor Works

An interactive animation showing how a Touch Sensor acts as an electrical switch to create a flow of electricity the EV3 detects.



Touch Sensor

Touch 4 : Forward until Touch

Forward until Touch

Topics Covered

- Motor "On" Mode
- Forward Until Behavior

Check Your Understanding:

- What does a Move command do when its Mode is set to "On"?
 - Turn the motors on
 - Turn the motors on for a certain number of rotations
 - Turn the motors on until the Touch Sensor is triggered
 - Combines with the next block to make a special command
- What does a Move command do when its Mode is set to "Off"?
 - Turn the motors off
 - Waits for the Touch Sensor to be pressed
 - Wait for the Touch Sensor to be pressed, then turn the motors off
 - End the program

Try It!

Forward Until Release

The Wait - Touch block can wait for the sensor to be "Released" as well as "Pressed".

What happens if you set the Wait - Touch block to "Released" and run it with an empty box holding down the sensor?

Note: When setting up robot, have the obstacle place firmly against the Touch Sensor so that it keeps it pressed in, as shown below.

What happens?

The robot moves forward until the Touch Sensor is "Released", then stops.

Touch 4: Forward until Touch

This step introduces the Motor On and Motor Off modes of the Move Steering Block.

Motor On – Wait for Touch – Motor Off form a common pattern that makes the robot move forward until the Touch Sensor is pressed, then stop. This pattern is sometimes referred to as “Forward until”.

Try It!: Forward Until Release

Prompts students to try the “Released” setting for the Wait for Touch Block instead of “Pressed”. The robot will move until the Touch Sensor is NOT pressed.

Make sure the Touch Sensor starts pressed in, or the “Released” state will be detected immediately, and the robot will go nowhere (see “Already Pressed” on the previous page).



Touch Sensor

Touch 4 (continued)

Mini-Challenge: Four Walls

Challenges students to make the robot move to touch all four walls of an enclosed rectangular space.

The desired behavior can be thought of as “Forward until Touch (the wall), then turn” four times. Backing up prior to turning may help to avoid getting caught on the wall.

Students with prior programming experience may know that a Loop Block would be helpful here; it is up to you to decide whether it is allowed at this time.

Touch 5: Touch Sensor Review

Expanded explanations for both in-video programs and a sample solution for the mini-challenge can be found on this page.

Mini Challenge

Mini Challenge 1: Four Walls
Program your robot to touch all four walls of a room, using its Touch Sensor to know when it has reached each one.

Use a 4x4 game board and place robot in the middle. Program your robot to touch all four walls, using its Touch Sensor.

PROGRAM THE ROBOT MOVE AND TOUCH ALL FOUR WALLS, USING ITS TOUCH SENSOR. THE ROBOT CAN TOUCH THE WALLS IN ANY ORDER AS LONG AS IT MAKES CONTACT WITH THE TOUCH SENSOR.

[Click Show hint to add hints here] [Show hint](#)

[Previous](#) [Next](#)

Touch 5 : Touch Sensor Review

Program Review: Wait for Touch
The program shown below is a sample code for making your robot wait until touch sensor is pressed.

Hover on any block in the program to view a detailed explanation.

Program Review: Forward until Touch
The program shown below is a sample code for ‘Forward until Touch’ movement. The robot first starts its motor to move forward, and when the touch sensor is pressed, the robot stops the motor.

Hover on any block in the program to view a detailed explanation.

Program Review: Vacuum
The program shown below is a sample solution to the Vacuum Mini-Challenge.

The solution reuses a four-block pattern, marked as “Behavior Pattern” below. In the pattern, the robot moves forward until it touches the wall, then backs up and turns 90 degrees to face the next wall. Since this behavior works for all four walls of the room, the program simply copy-pastes that pattern three more times.

Hover on any block in the program to view a detailed explanation.

Copy of Behavior Pattern



Touch Sensor

Touch 6 : Arm Position Challenge

Challenge Review

Arm Position Challenge

TOUCH SENSOR ATTACHMENT (Modified)

Robot Educator Building Instructions Driving Base 1/16 + Touch Sensor - Driving B... CUSTOM BUILT ATTACHMENT Where can I find the instructions?

In this challenge, you will program your EV3 robot to raise its arm when pressing the Up button on the EV3, then retrieving a container and bringing it back to the starting location.

1. PRESSING THE UP BUTTON ON EV3 RAISES ARM TO UP POSITION
2. ROBOT MOVES FORWARD TWO (2) ROTATIONS TO CONTAINER
3. USING ARMS, ROBOT TAKES CONTAINER BACK TO START

Challenge PDF [armPosition_challenge.pdf] Download

Previous Complete

Copyright©2012 Carnegie Mellon Robotics Academy. All rights reserved.

Arm Position Challenge

This step lays out the details for the Arm Position Challenge.

This challenge involves more physical construction than programming, as students will need to find a way to position the Touch Sensor so that it can detect the Arm reaching the top of its motion.

Students should be allowed limited freedom to modify the robot's arm or add extensions during this Challenge, but should not need to modify or disassemble the driving base.

A concept picture of one possible layout is shown on the page, but no specific building instructions are provided: students should work to complete a design on their own.

Arm Position Challenge PDF

Detailed measurements for the board layout, as well as instructions for setting it up, rule details, and hints for solving the challenge can be found in this printable document.

Introduction to Programming
LEGO® MINDSTORM® EV3

CHAPTER CHALLENGE

CHAPTER 3: Arm Position Challenge

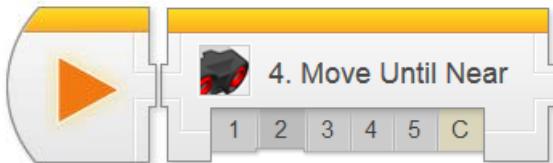
In this challenge, you will program the robot's arm to move into the "Up" position when the "Up" button on the EV3 is pressed, no matter where the arm started. The robot will then move forward two (2) rotations to pick up a cargo container, and bring it back to the starting location.

Rules and Procedures:

The robot's arm will be moved to a random position before running.
You have parts available to you to modify the Touch Sensor parts mounting on that it can detect when the EV3's arm is in the "up" position.
Try to do this with as few changes as possible.
When the Up button is pressed, the robot must move forward two rotations to the "Up" position [1], move forward two rotations to the cargo [2], and bring it back to the original starting position [3].

Hints:

- The trigger area on the Touch Sensor is small. You will probably need to build a "bumper" or "extender" on the end of the Touch Sensor to make it detect the arm more reliably.
- The EV3 core set includes two Touch Sensors. You can use one sensor for detecting the arm in the "Up" position, and the other to detect the box.
- You can also make a test run, then calculate "how many times as far" you need to move or turn to get the amount of movement you want, compared to a test run.



Introduction to the Ultrasonic Sensor Unit

Sensors > Move Until Near Chapter

In Move Until Near, students program a robot to wait until its Ultrasonic Sensor detects an object within a certain “threshold” distance before proceeding with other commands. This is combined with Motors On and Motors Off to produce a Forward Until behavior parallel to the one constructed using Touch in the previous chapter.

Key Concepts: Ultrasonic Sensor, Threshold Value, Forward Until Pattern

► **Ultrasonic 1: Introduction to Sensors**

Introduces the real-world robot (Hexarotor), and the challenge modeled after it (Maze Challenge)

► **Ultrasonic 2: Robot Config**

Contains building and setup instructions for the rest of the chapter

► **Ultrasonic 3: Wait for Near**

Uses the Wait for Near Block to make a robot wait for the Ultrasonic Sensor to detect an object closer than the threshold value before continuing

► **Ultrasonic 4: Forward until Near**

Combines Motor On and Motor Off commands with Wait for Near to make the robot move forward until the Ultrasonic Sensor detects something nearby

► **Ultrasonic 5: Review**

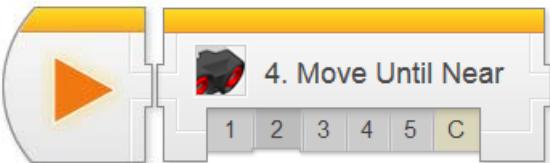
Explains sample solutions to mini-challenges from this chapter

► **Maze Challenge**

Requires students to program an Ultrasonic Sensor-based solution to a maze, detecting distances from key walls to known when to turn.

Hints:

- All of the chapters in the Sensors unit follow the general pattern laid out in this chapter: Introduction, Wait for (Sensor), Forward until (Sensor), Challenge.



Ultrasonic Sensor

Ultrasonic 1 : Introduction

Introduces the Autonomous Hexarotor and the Maze Challenge.

Autonomous Hexarotor
2013

Topics Covered

- Autonomous Hexarotor
- Challenge overview

Check Your Understanding:

Q 1. Why might the Rangefinders (Laser or Ultrasonic) be preferred over Touch Sensors for detecting walls and obstacles?

- Hitting wall could damage the wall
- Hitting a wall could damage the robot
- It is quicker to detect obstacles at a distance
- All of the above

Next

Copyright©2012 Carnegie Mellon Robotics Academy. All rights reserved.

Ultrasonic 2 : Robot Config

Ultrasonic Sensor Configuration
For this chapter, the Ultrasonic Sensor is required to complete sections of this chapter as well for the final challenge.

ULTRASONIC SENSOR ATTACHMENT

ULTRASONIC SENSOR PLACEMENT
PLEASE REFER TO THE BUILDING INSTRUCTIONS CONTAINED IN THE EV3 SOFTWARE

Driving Base 1/18 + Ultrasonic Sensor - Drive... 7/18

Robot Educator Building Instructions
Robot Educator Building Instructions

Where can I find the instructions?

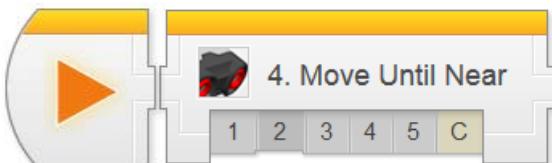
Previous Next

Copyright©2012 Carnegie Mellon Robotics Academy. All rights reserved.

Ultrasonic 2: Robot Config

This chapter uses the driving base model, plus a forward-facing Ultrasonic Sensor.

Sensors from the previous chapter can be left on or removed as convenient, as long as they do not interfere with the Ultrasonic Sensor (i.e. are not within its “cone” of view).



Ultrasonic Sensor

Ultrasonic 3 : Turning in Place

This step introduces the Ultrasonic Sensor and Wait for Near Blocks. The Sound Block is used also, as a way of indicating when the sensor has been triggered.

Forward Until Near

Check Your Understanding:

- What does the Wait Block wait for before playing the "Hello" sound?
 - The Ultrasonic Sensor to detect an object less than 50 cm away
 - The Ultrasonic Sensor to detect an object more than 50 cm away
 - The robot to travel less than 50 cm
 - The Ultrasonic Sensor's reading to change by up to 50 cm
- How do you select the sound file the robot plays?
 - By clicking in the upper-right "File Name" blank on the block
 - By speaking into the speakers of the EV3
 - By adding a File Block

Mini Challenge

Mini Challenge 1: Threshold Value

The Wait - Ultrasonic Sensor Block uses a "Threshold" to define what it is waiting for. Rather than look for a specific value like 1cm or 200cm, it sets a "cutoff" value that divides all the possible Ultrasonic Sensor values into two categories:

- If the distance value is above the Threshold, it is considered "Far"
- If the distance value is below the Threshold, it is considered "Near"

This way, the Wait Block does not have to worry about the difference between an object at 29cm and an object at 30cm; it only has to worry about whether the value is above or below the Threshold.

Change the Wait Block's Threshold value so that the alarm only sounds if someone passes within 10cm of the sensor, then change it again so that it will go off if anyone comes within 100cm of the sensor.

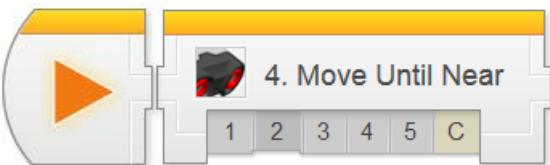
(Click Show hint to add hints here)

Show hint

Mini-Challenge: Threshold Value

Challenges students to alter the Threshold value on their Wait for Near Block to trigger at 10 cm, and at 100 cm.

Thresholds are a key concept in robotics programming, because they allow programs to easily make decisions based on sensor readings that range over hundreds of possible values (1 cm, 2 cm, 3 cm...). Rather than write hundreds of different responses, the robot simply responds to the number being above or below the Threshold cutoff.



Ultrasonic Sensor

Try It! 1 **Missing Object Alarm (Wait for Far)**

What does the Wait Block do if you set it to wait for a value Greater Than the Threshold instead of Less Than? Change the Wait Block's "Compare Type" setting to Greater Than (2).

Place an object in front of the robot, and download and run the program. Now, try moving the object and see what happens.

Optional Activities

Did You Know?

Did you know?
How the Ultrasonic Sensor Works

The Ultrasonic Sensor uses the speed that sound waves travel to measure distance to an object. The sensor has two openings on its front, one opening emits ultrasonic waves, while the other receives them. The Ultrasonic Sensor measures distance by timing how long it takes for an ultrasonic wave sent out by the emitter to bounce off an object and come back to the receiver.

Try It! 2 **Sound Sentences**

You can have multiple sounds play one after another to form sentences. Try adding a second Sound Block to your program so it says "Object Detected" instead of just "Hello". "Object" and "Detected" can be found as separate sound files in the "Information" folder. How would you get the EV3 to say them one after another?

Try It! 3 **Sensor Change Mode**

In addition to "Comparing" the value of the sensor against the Threshold, the Wait Block can also look at the amount the sensor value has **changed** since the command started.

Ultrasonic 3 (continued)

Try It!: Missing Object Alarm

Prompts students to try using Greater Than Threshold instead of Less Than Threshold. This is analogous to Wait for Release instead of Wait for Pressed.

[Optional] Did You Know?: How the Ultrasonic Sensor Works

A video animation showing how an Ultrasonic Sensor uses sonar to calculate the distance to an object.

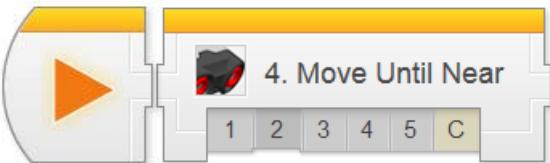
[Optional] Try It!: Sound Sentences

Adding sounds within your program provide excellent clues that alert you that a behavior has just completed.

Focusing on the Sound Block, this Try It! prompts students to string together multiple blocks (saying different words) into a sentence.

[Optional] Try It!: Sensor Change Mode

Prompts students to try using the Change mode to wait for a change in Ultrasonic values, rather than a value above or below a particular threshold (as it does in Compare mode).



Ultrasonic Sensor

Ultrasonic 4 : Forward until Near

This step guides students through a Forward Until behavior that uses the Ultrasonic Sensor to stop when the robot gets close to a wall (Ultrasonic Sensor value is below a certain distance threshold).

This is analogous to Forward Until Pressed in the Touch Sensor unit.

Check Your Understanding:

Q 1. If the robot is facing a wall, it will move until...

- The Ultrasonic Sensor has traveled 50 cm
- The Ultrasonic Sensor is 50 cm from the wall
- The robot moves 50 cm total
- The robot detects an obstacle beyond 50 cm away

Mini Challenge

Mini Challenge 1: Backward Until Far

The program above is Forward Until Near, meaning the robot will move forward until it detects a wall or object within the set threshold. Now, perform the opposite: Create a program that makes the robot make backwards from a wall or object until it is beyond the set threshold.

• Place the robot near a wall or object, facing it.
• Program the robot to back away from it until it is 30 cm away.

1. PLACE ROBOT NEAR A WALL, FACING IT
2. PROGRAM ROBOT TO BACK AWAY UNTIL WALL IS 30CM AWAY

[Click Show hint to add hints here] Show hint

Previous Next

Copyright©2012 Carnegie Mellon Robotics Academy. All rights reserved.

Ultrasonic 4: Forward Until Near

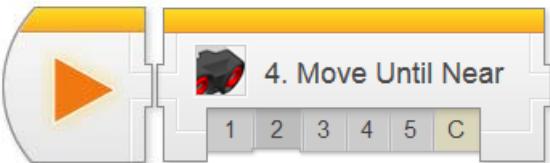
This step guides students through a Forward Until behavior that uses the Ultrasonic Sensor to stop when the robot gets close to a wall (Ultrasonic Sensor value is below a certain distance threshold).

This is analogous to Forward Until Pressed in the Touch Sensor unit.

Mini-Challenge: Backward Until Far

Challenges students to adapt their behavior to move backwards until the robot is a certain distance away from an obstacle.

This activity is directly analogous to the “Forward Until Release” mini-challenge in the previous chapter, except that the robot needs to back up so that it gets farther from the wall (rather than continuing to get nearer).



Ultrasonic Sensor

Ultrasonic 5: Ultrasonic Review

Expanded explanations for both in-video programs can be found on this page.

Ultrasonic 5 : Ultrasonic Sensor Review

Program Review: Wait for Touch

- The program shown below is a sample code for making your robot wait until touch sensor is pressed.

Hover on any block in the program to view a detailed explanation.

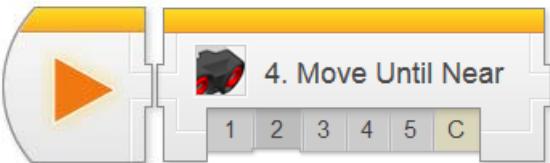
Program Review: Forward until Touch

- The program shown below is a sample code for 'Forward until Touch' movement. The robot first starts its motor to move forward, and when the touch sensor is pressed, the robot stops the motor.

Hover on any block in the program to view a detailed explanation.

Previous Next

Copyright 2012 Carnegie Mellon Robotics Academy. All rights reserved.



Ultrasonic Sensor

Ultrasonic 6 : Maze Challenge

Challenge Review

Maze Challenge

In this challenge, you will program your EV3 robot to move from the starting area through a maze with tall vertical walls. Use the Ultrasonic Sensor to navigate through the maze without touching any walls and ultimately reaching the goal zone regardless of what the distances were between the walls.

1. PROGRAM THE ROBOT TO NAVIGATE ITSELF THROUGH THE MAZE.
2. THE WALLS CAN BE MOVED NEARER OR FARTHER BETWEEN RUNS.

USE BOOKS OR BINDERS TO RECREATE THE MAZE
WALLS CAN BE MOVED. HOWEVER THE MAIN PATH
WILL NEVER CHANGE

Challenge PDF
[Download](#)

Previous **Complete!**

Maze Challenge

This step lays out the details for the Maze Challenge. Students should work in their teams to complete the challenge objectives.

The route shown in the picture is the only real route to the goal. However, certain walls are allowed to move, or even be removed (see PDF instructions).

The robot must use the “guaranteed” walls to find its way through the maze without touching any walls.

EV3-VT CHALLENGE

CHAPTER 3: Maze Challenge

In this challenge, you will program the EV3 robot to move from its starting area through a maze with tall, vertical walls. Walls in the maze can be adjusted to be nearer or farther between each run, but the general path must remain unchanged. Make use of the Ultrasonic Sensor and be sure to not touch any walls along the way.

Rules and Procedures:

- Recreate the maze with tall objects, such as books or binders.
- The robot must start entirely within the start box (1), as well as stopping entirely inside the end box (2).
- Walls marked by gray lines can move slightly.
- Walls marked by dark black lines cannot be moved.
- The robot must not touch any walls as it navigates throughout the maze.

Hints:

- The patterns of turns cannot change, so you don't need to worry about using a sensor (other than Flicker) during turns.
- Look carefully at what distances in the maze are guaranteed not to change. Try to use those distances in combination with the Ultrasonic Sensor to orient yourself.

Maze Challenge PDF

Detailed measurements for the board layout, as well as instructions for setting it up, rule details, and hints for solving the challenge can be found in this printable document.



Introduction to the Gyro Sensor Unit

Sensors > Turn for Angle Chapter

In Turn for Angle, students program a robot to turn until its Gyro Sensor detects a heading change greater than a certain “threshold”. This is the same pattern used with Forward Until Touch and Forward Until Near, but with turning.

Due to the nature of the Gyro Sensor, and of turning motion in general, a small amount of “overshoot” in the turn will occur. The programming in this lesson is broken into two steps to explicitly address and accommodate this phenomenon.

Key Concepts: Gyro Sensor, Accommodating Sensor Error

► **Gyro 1: Introduction to Sensors**

Introduces the real-world robot (Autonomous Golf Course Mower), and the challenge modeled after it (Mower Challenge)

► **Gyro 2: Robot Config**

Contains building and setup instructions for the rest of the chapter

► **Gyro 3: Turn to Angle (Part 1)**

Uses a variant of the Forward Until Pattern that turns instead of moving straight, and uses the Gyro Sensor to detect the change in heading. Program will work, but not be as accurate as desired.

► **Gyro 4: Turn to Angle (Part 2)**

Discusses physical factors that contribute to the inaccuracy of the original motion, and strategies to mitigate the issue.

► **Gyro 5: Review**

Explains sample solutions to mini-challenges from this chapter

► **Mower Challenge**

Requires students to program a Gyro Sensor-based behavior that will drive the robot over all parts of a rectangular surface, even if it loses traction partway through a turn.

Hints:

- The Gyro Sensor, like all real-world devices, is imperfect. This lesson uses that imperfection as a learning opportunity.



Gyro Sensor

Gyro 1 : Introduction to Golf Course Mower

Autonomous Golf Course Mower
Gyro Sensor Challenge

Topics Covered

- Autonomous Hexarotor
- Challenge overview

Check Your Understanding:

- What does a gyro sensor help the robot to do?
 - Move a precise distance
 - Turn more precisely
 - Use GPS
 - None of the above
- Why does the autonomous mower use a gyro sensor if it already has GPS?
 - Gyro sensor is more accurate than GPS
 - GPS is sometimes blocked or unavailable
 - GPS can be slow sometimes
 - Robot moves faster using gyro sensor
- Why should your robot use a gyro sensor even if it already has rotation sensors?
 - Wheels sometimes slip and lose accuracy
 - Motor/encoder rotation sensors cannot be used to detect turns/rotation

Gyro 2 : Robot Config

Gyro Sensor Configuration

The Gyro Sensor is required to complete sections of this chapter as well for the final challenge.

GYRO SENSOR ATTACHMENT

GYRO SENSOR PLACEMENT
PLEASE REFER TO THE BUILDING INSTRUCTIONS CONTAINED IN THE EV3 SOFTWARE

Building Instructions

- Driving Base: 1/18
- + Gyro Sensor - Driving Base: 0/19

Where can I find the instructions?

Gyro 1: Introduction

Introduces the Autonomous Golf Course Mower and Challenge

Gyro 2: Robot Config

This chapter uses the driving base model, plus a Gyro Sensor.

Sensors from the previous chapter can be left on or removed as convenient.



Gyro Sensor

Gyro 3 : Turn for Angle (Part 1)

This step introduces Turn for Angle in the way that it would be expected to work, following the Forward Until pattern.

Due to a number of unavoidable physical factors, the robot will noticeably “overturn” the target angle. This phenomenon is addressed in the next step.

Check Your Understanding:

- Q** 1. Because of the way it is attached to the robot, the Gyro Sensor measures:
 - The amount the robot's body turns
 - The amount the robot's wheels turn
 - The amount the robot is "tipping" forward or backward
 - When an oscilloscope is nearby
- Q** 2. When the robot actually ran, what happened?
 - The robot turned its body exactly 90 degrees
 - The robot turned its body slightly more than 90 degrees
 - The robot moved forward 90 degrees
 - The robot spun in place forever

Previous **Next**

Copyright ©2012 Carnegie Mellon Robotics Academy. All rights reserved.

Gyro 4 : Turn for Angle (Part 2)

This step discusses the reasons for the seemingly strange behavior of the Gyro Sensor in the previous step, and how this can lead to a workaround (and what a workaround is, compared to a solution).

Check Your Understanding:

- Q** 1. Which of the following factors contributes to the “overturning” problem?
 - Sensor accuracy limitations
 - Delay in sensing and signal transmission
 - Physical momentum
 - All of the above
- Q** 2. Which of the following workarounds can help to reduce the “overturning” problem?
 - Replace the Gyro sensor
 - Telling the robot to wait until a value that comes “before” the one you actually want
 - Use a different numbered port
 - Press the cancel button as soon as it completed its turn
- Q** 3. What is the difference between a workaround and a solution?
 - A workaround is preferred over a solution
 - A solution removes the source of the problem, while a workaround only reduces its effects



Gyro Sensor

Try it!

Try It! 1 Left Turns

Does the same Wait for Gyro block work for left turns?
Try changing your program to turn 90 degrees **to the left** instead.
What happens?

What happens?

Mini Challenge

Mini Challenge 1: Rectangle Box
Program and make the robot complete a full lap around a rectangular box, using the Gyro Sensor to control all of its turns.

[Click Show hint to add hints here]

Show hint

Optional Activities

Did You Know?

Did you know?
How the Gyro Sensor Works

The EV3 Gyro Sensor is a **MEMS** Sensor (Micro-ElectroMechanical System)

Copyright ©2012 Carnegie Mellon Robotics Academy. All rights reserved.

Gyro 4 (continued)

Try It: Left Turns

Prompts students to try using the Gyro Sensor to perform a left turn.

Since the Wait For Gyro Block is in Change mode, left and right turns both work – going 90 degrees in either direction is still a “change”. This is not true in Compare mode, however, as Compare mode uses compass headings that will go negative if the robot turns left (and thus reach -90, not 90).

Mini-Challenge: Rectangle Box

Challenges students to make the robot run around a rectangular (or square) object, using the Gyro Sensor to make each turn.

As with most mini-challenges, this task is very closely related to what students will be expected to do during the Challenge.

[Optional] How the Gyro Sensor Works

Explanation of the physical principle of operation of the Gyro Sensor.



Gyro Sensor

A screenshot of a web-based review page for the "Gyro 5 : Gyro Sensor Review". The page has a yellow header bar. Below it, the title "Gyro 5 : Gyro Sensor Review" is displayed. A "Program Review: Turn for Angle (60 degrees)" section contains a list of bullet points explaining the program's logic. Below this is a "Program Review: Box challenge" section with another list of bullet points. At the bottom of the page are "Previous" and "Next" navigation buttons.

Gyro 5: Review

Sample solutions and explanations for both mini-challenges can be found on this page.

A screenshot of a web-based review page for the "Box challenge". The page has a yellow header bar. Below it, the title "Box challenge" is displayed. A "Program Review: Box challenge" section contains a list of bullet points explaining the program's logic. At the bottom of the page are "Previous" and "Next" navigation buttons.



Gyro Sensor

Gyro 6 : Mower Challenge

Challenge Review

Mower Challenge

ERASER ATTACHMENT*
*ACTUAL DESIGN MAY VARY

SCOOPT ATTACHMENT*
*ACTUAL DESIGN MAY VARY

In this challenge, you will program your EV3 robot to erase or clear the entire gameboard of either markings or parts. The robot is able to move freely in straight lines, using any method you want. However, there are three mud zones marked on the game board. When turning in one of these areas, the robot must be picked up by hand, and placed back down.

ROBOT MUST CLEAR BOARD OF MARKINGS OR PARTS.
THE GAMEBOARD HAS THREE (3) MUD ZONES AT SPECIFIED LOCATIONS.

WHEN TURNING ON A MUD ZONE,
THE ROBOT MUST BE PICKED UP AND PLACED BACK DOWN.

Notes:

Challenge PDF
[mower_challenge.pdf]

Download

Previous

Complete!

Copyright©2012 Carnegie Mellon Robotics Academy. All rights reserved.

Mower Challenge

This step lays out the details for the Mower Challenge. Students should work in their teams to complete the challenge objectives.

The “mud” zones (red squares) are designed to make Wheel Rotations-based solutions impossible. The robot should be picked up, held briefly in the air (with the wheels still spinning), then placed back down when the robot turns on top of a mud patch.

The mud zones are positioned so the robot will have to turn while on top of at least one.

If you have a game board surface that erases cleanly, the eraser version of the board may work better. If not, use the loose-parts version, but make allowances for a large number of parts rolling onto the floor.

Mower Challenge PDF

Detailed measurements for the board layout, as well as instructions for setting it up, rule details, and hints for solving the challenge can be found in this printable document.

EV3-VT CHALLENGE

CHAPTER 5: Mower Challenge

In this challenge, you will program your EV3 robot to erase or clear the entire gameboard of either markings or parts. The robot is able to move freely in straight lines, using any method you want. However, there are three mud zones marked on the game board. When turning in one of these areas, the robot must be picked up by hand, and placed back down.

Rules and Procedures:

- The robot must clear the board of marks or parts to complete its mission.
- If any part of the robot is in the mud zone (red squares) at any point during a turn, it must be picked up and placed back down, as close to the same spot and facing as possible.
- Use overlapping paths to compensate for sensor inaccuracies.

Hints:

- Because the robot's wheels continue to spin in the air when it is picked up, Rotations or Time will not be reliable for turns in the mud.
- The Gyro Sensor responds only to the robot's body turning, and is unaffected by interruptions like being picked up.
- It is unlikely that the eraser (or scoop) will perform perfectly, especially near the edges of its reach. Plan your robot's course accordingly.
- Use an adjustment factor to compensate for the fact that the robot won't see 90 degrees until it is past 90 degrees.



Introduction to the Color Sensor Unit

Sensors > Move Until Color Chapter

In Move Until Color, students program a robot to wait until its Color Sensor detects an object of a certain color before proceeding with other commands. This is combined with Motors On and Motors Off to produce a Forward Until behavior parallel to the ones using Touch and Ultrasonic Sonar in the previous chapter.

Key Concepts: Color Sensor, Forward Until Pattern

► **Color 1: Introduction to Sensors**

Introduces the real-world robot (Autonomous Motor Vehicle), and the challenge modeled after it (Traffic Light Challenge)

► **Color 2: Robot Configuration**

Contains building and setup instructions for the rest of the chapter

► **Color 3: Wait for Green**

Uses the Wait for Color Block to make a robot wait for the Color Sensor to detect a green object in front of it before moving

► **Color 4: Forward until Red**

Combines Motor On and Motor Off commands with Wait for Color to make the robot move forward until the Color Sensor detects a red object

► **Color 5: Review**

Explains sample solutions to mini-challenges from this chapter

► **Traffic Lights Challenge**

Requires students to program a robot to move through three traffic signals, each of which may be red or green at random

Hints:

- The main programming pattern in this chapter is very similar to the previous chapters, but the logical complexity of the Challenge solution is higher. Students must continue improving their understanding of the ways in which behaviors relate, to produce the best (and simplest) possible solution



Color Sensor

Color 1 : Introduction to Autonomous Vehicle

Autonomous Motor Vehicles
Victorville, CA 2007

Topics Covered

- Autonomous Motor Vehicles
- Challenge overview

Check Your Understanding:

Q 1. What are some of the challenges a self-driving car must overcome?

- Finding a path to the destination
- Following the road
- Obedying traffic laws and signals
- Avoiding other vehicles
- All of the above

Q 2. What will the robot need to detect with the Color Sensor?

- The speed of the robot
- Distance from the vehicle in front of the robot
- Width of the road
- Color of a traffic light

Next

Copyright©2012 Carnegie Mellon Robotics Academy. All rights reserved.

Color 1: Introduction

Introduces the Autonomous Motor Vehicle and the Traffic Light Challenge.

Color 2 : Robot Config

Color Sensor Configuration

The Color Sensor is required to complete sections of this chapter as well for the final challenge.

COLOR SENSOR ATTACHMENT

COLOR SENSOR PLACEMENT
PLEASE REFER TO THE BUILDING INSTRUCTIONS CONTAINED IN THE EVO SOFTWARE

Robot Educator Building Instructions Driving Base 1/16 + Colour Sensor Forward 5/16

Where can I find the instructions?

Color 2: Robot Config

For the video portion of this chapter, the Color Sensor will be mounted in the default location. It will need to be moved to the arm for the Challenge, as it will collide with the traffic signal otherwise.

If this causes an issue, you can change the rules of the Challenge to permit the student to move the traffic signal out of the way rather than moving the sensor to avoid it.



Color Sensor

Color 3 : Wait for Green

This step introduces the Wait for Color Block. This step is structurally similar to the other Wait For behaviors – the Wait For Color Block will wait until ANY ONE of the selected colors is detected.

Topics Covered

- Wait for Color Block

Wait for Green

Check Your Understanding:

- What does this program do?
 - Wait for the Color Sensor to see a Rod object, then move forward
 - Wait for the Color Sensor to see a Green object, then move forward
 - Wait until an object is moved out of the way, then move forward
 - Move forward until it sees a Green object
- When multiple colors are checked in the "Set of Colors" area, what will the Wait Block do?
 - Wait for ANY of the colors to be seen
 - Wait for ALL of the colors to be seen at the same time
 - Wait for ALL of the colors to be seen at least once each
 - Wait for ALL of the colors to be seen in the order indicated by the numbers

Try It!

The EV3 Color Sensor can detect 7 different colors, plus the absence of color. Each of these 8 colors is labeled with a different number.

Try It! 1 | No Color

What does the No Color "color" mean in the Set of Colors menu?

Change your program so that it waits for "No Color" (Option 0). Make sure you unselect all the other colors.

Try It! 2 | Port View: Color Sensor Values

You can see the Number value of the currently detected color directly on the EV3's viewscreen, in the Port View Mode.

1. Use the Left and Right Buttons on the EV3 to navigate to the EV3 Apps menu (), and press the Enter Button to select "Port View Mode".

The 8 blocks at the top and bottom of the screen represent the 8 ports on the EV3.

- Motor Rotation Sensor values are displayed across the top.
- Sensor Values are displayed across the bottom, depending on what is plugged in

What happens?

Motor Rotation Sensor values

Color 3: Wait for Green

This step introduces the Wait for Color Block. This step is structurally similar to the other Wait For behaviors – the Wait For Color Block will wait until ANY ONE of the selected colors is detected.

Try It! 1: No Color

The “No Color” option represents the state where the Color Sensor is not getting a reading from a surface in front of it.

Try It! 2: Port View Color Sensor Values

The Port View mode on the EV3 on-screen menus don’t call colors by name. Instead, they use the numeric codes shown next to the numbers in the checklist.



Color Sensor

Color 3 (continued)

1. Point the Color Sensor directly at the Red side of the Color Crate. What value do you see?
[What happens?]

2. What value should you see if the Color Sensor is pointed at the Blue side of the Color Crate? Try it!
[What happens?]

3. Point the Color Sensor away from the Color Crate and any strong lights. What reading does it give? Try it!
[What happens?]

Mini Challenge

Mini Challenge 1: Railroad Crossing
Instead of red and green lights, some traffic signals simply read signs that raise and lower in the path of traffic.

Program your robot so that instead of waiting for a green light, it waits for the red stop sign to be taken away.

Stop

Go!

Optional Activities

Did You Know?

Did you know? How the Color Sensor Works

Previous **Next**

Copyright©2012 Carnegie Mellon Robotics Academy. All rights reserved.

Mini-Challenge: Railroad Crossing

Instead of waiting for Green to go, Wait for “No Color”.

[Optional] How the Color Sensor Works

The Color Sensor is actually three different sensors: a Red Sensor, a Blue Sensor, and a Green Sensor. By measuring the relative amounts of red, green, and blue light that a surface reflects, the sensor can tell the color of the surface.



Color Sensor

Color 4 : Forward until Red

Topics Covered

- Color Sensor
- Forward Until Red

Forward until Red

Check Your Understanding:

- Q 1. What does this program do when run?
- Move forward when the robot sees a red object
 - Move forward until the robot sees a red object
 - Move backward when the robot sees a red object
 - Move faster when the robot sees a red object

Mini Challenge



In addition to signs and lights, self-driving cars also need to obey pavement markings, like this stop line telling where to stop.

Modify your Color Sensor attachment, so that it faces downward, and program your robot to drive forward until it reaches the line.

Color 4: Forward until Red

This step adds the Motor On and Motor Off commands to make a Forward Until behavior like the ones from previous chapters.

Mini-Challenge: Forward to Stop Line

Face the Light Sensor down at the ground to detect colored lines on the table surface. This is a common use of the Color Sensor in robotics competitions.

COLOR SENSOR (DOWN) ATTACHMENT*

*ACTUAL DESIGN MAY VARY

COLOR SENSOR (DOWN) PLACEMENT

MODIFY THE COLOR SENSOR SO THAT IT POINTS DOWN, IN ORDER TO DETECT LINES ON SURFACES

Driving Base 1/16

Colour Sensor Down - Dr... 4/16

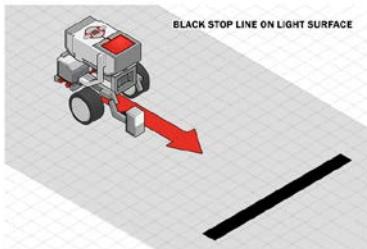
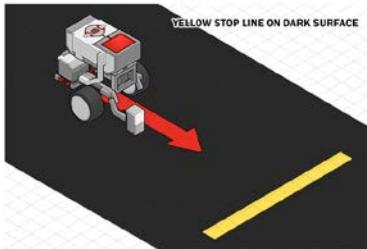
Robot Educator Building Instructions

Robot Educator Building Instructions

Where can I find the instructions?



Color Sensor



[Click Show hint to add hints here]

Show hint



Color 5 : Color Sensor Review

Program Review: Wait for Green

- The program shown below is sample code for making your robot wait until it sees the color green.

Hover on any block in the program to view a detailed explanation.

Program Review: Forward to Stop Line

- The program shown below is sample code for 'Forward to Stop Line' mini-challenge. The program assumes that the robot is running on a white table, looking for a black line.
- First, the robot starts its motor to move forward, and when the color sensor sees the color black, the robot stops the motor.

Hover on any block in the program to view a detailed explanation.

Previous

Next

Color 4 (continued)

Now try the challenge on other surfaces with other colors!

Color 5: Color Sensor Review

Expanded explanations for the Wait for Green program and the Forward to Stop Line program can be found on this page.



Color Sensor

Color 6 : Traffic Signal Challenge

Challenge Review

Traffic Lights Challenge

COLOR ARM ATTACHMENT*

COLOR ARM PLACEMENT: ATTACH THE COLOR SENSOR SO THAT IT CAN REACH COLOR OR THE ELEVATED TRAFFIC LIGHTS. PLACING THE SENSOR ON THE ARM IS ONE POSSIBLE METHOD.

Driving Base

Building Instructions

Robot Educator

Where can I find the instructions?

In this challenge, you will program your EV3 robot to go through three different intersections, each of which has a traffic signal. The traffic signal, which can be either the colored block or the red/green card, is held by hand at a set height. Unlike a camera, the detection range of the Color Sensor is short, so you will need to modify its placement on the robot so that it can see the traffic signal and react appropriately.

Bonus: The robot does not need to stop on its own after passing through all three intersections (it can be stopped by hand).

HOLD THE TRAFFIC SIGNAL AT A SET HEIGHT AND HAVE THE ROBOT REACT ACCORDINGLY DEPENDING ON THE SIGNAL

Write the program so that it DOES always stop after going through the third intersection. This will require the use of a Switch (see next section).

Challenge PDF [traffic_challenge.pdf]

Download

Previous

Complete!

Copyright©2012 Carnegie Mellon Robotics Academy. All rights reserved.

Traffic Signal Challenge

This step lays out the details for the Traffic Signal Challenge. Students should work in their teams to complete the challenge objectives.

Teams will need to reposition the Color Sensor to the robot's arm so that it can reach high enough to see the traffic signals, and also move the sensor out of the way afterward. If this is inconvenient, modify the challenge rules to allow the signal to be moved out of the way.

Traffic Signal Challenge PDF

Detailed measurements for the board layout, as well as instructions for setting it up, rule details, and hints for solving the challenge can be found in this printable document.

Introduction to Programming
LEGO® MINDSTORM® EV3

CHAPTER CHALLENGE

CHAPTER 6: Traffic Signal Challenge

In this challenge, you will program your EV3 robot to go through three different intersections, each of which has a traffic signal. The traffic signal, which can be either the colored block or the red/green card, is held by hand at a set height. Unlike a camera, the detection range of the Color Sensor is short, so you will need to modify its placement on the robot so that it can see the traffic signal and react appropriately.

Rules and Procedures

The traffic signals are represented by holding the colored block with the signal side downward and toward the robot at a 45-degree angle.

- You can also use red and green colored sheets of paper, which may be easier to hold at the correct height.
- The traffic signal color for each intersection is determined by flipping three coins before the run. Heads = Green, Tails = Red.
- The robot **MUST** stop if the light is red, and **MUST NOT** stop if the light is green.
- After the robot successfully gets through all three intersections, the robot can be stopped by hand.

Hints:

- Since you only have access to Wait commands, try to break the robot's behavior into stages. What is the robot waiting to see at each stage of its movement? What should its motors be doing during those stages?
- If the robot is already moving, is there any point in waiting for Green? If the robot is stopped, is there any point in waiting for Red?
- If the robot sees a red light, don't forget that it needs to wait for the actual Green light before proceeding!



Introduction to Robot Decisions - Loops

Decisions > Loops Chapter

In the Decisions Unit, students begin to work with Program Flow more directly. Loops give programmers the ability to repeat commands; many students already know this. However, the real power of Loops lies with the capability to decide whether to repeat or not.

Key Concepts: Program Flow, Loops, Conditional Loops

► **Loops 1: Introduction to the Container Handling Robot**

Introduces the real-world robot (Container Handling Robot), and the challenge modeled after it (Container Handling Challenge)

► **Loops 2: Robot Configuration**

Contains building and setup instructions for the rest of the chapter

► **Loops 3: Looped Movement**

Uses a simple (infinite) loop to make the robot move back and forth repeatedly

► **Loops 4: Loop with Count Control**

Sets the Loop to repeat only a set number of times

► **Loops 5: Loop with Count Control**

Uses a Sensor to decide whether the Loop repeats or not at the end of each pass

► **Loops 6: Review**

Explains sample solutions to mini-challenges from this chapter

► **Container Handling Challenge**

Requires students to use loops and sensor control to program a robot to pick up a number of objects and move them back to a starting zone

Hints:

- The concept of repeating is simple, but it is important to begin building a broader mental model of Program Flow now so students can understand how Loops relate to Switches and Switch-Loops in the next chapters

Decisions - Loops



Loops 1 : Introduction to Container Handling Robot

Container Handling Robot
2004

Topics Covered

- Autonomous Motor Vehicles
- Challenge overview

Check Your Understanding:

Q 1. Repetitive tasks can be accomplished efficiently by...
 Writing very long programs
 Writing multiple programs
 Repeating behaviors within a program
 Manually controlling the robot

Next

Copyright©2012 Carnegie Mellon Robotics Academy. All rights reserved.

Loops 1: Introduction

Introduces the Container Handling Robot, the Container Handling Challenge, and Loops.

Loops 2 : Robot Config

Ultrasonic Sensor Configuration
The Ultrasonic Sensor is required to complete sections of this chapter.

ULTRASONIC SENSOR ATTACHMENT

ULTRASONIC SENSOR PLACEMENT
PLEASE REFER TO THE BUILDING INSTRUCTIONS CONTAINED IN THE EV3 SOFTWARE

Driving Base 1/18 + Ultrasonic Sensor - Driv... 7/18

Robot Educator Building Instructions

Robot Educator Building Instructions

Where can I find the instructions?

Previous Next

Copyright©2012 Carnegie Mellon Robotics Academy. All rights reserved.

Loops 2: Robot Config

The robot for this chapter uses the Ultrasonic Sensor, and will eventually add the Color Sensor in the Challenge.



Decisions - Loops

Loops 3 : Looped Movement

Topics Covered: Loops

Check Your Understanding:

- What does the Loop do?
 - Send the program "flow" back to an earlier point in the program, causing it to repeat some instructions
 - Choose between two different possible sets of commands to run
 - Repeat a branching decision quickly, to enable continuous control of the robot
 - Run all the programs on the robot in a continuous cycle
- How do you add a Loop to a program?
 - Click the Loop Block, then drag a box around the commands you want to put inside the Loop
 - Select the blocks you want to put inside the Loop, then right-click and choose "Place in Loop"
 - Right-click in an empty area and choose "Make Loop" from the menu that appears
 - Drag a Loop into the program, then drag commands into it

Mini Challenge

Mini Challenge 1: Square Lap 1
Program a Looped behavior that makes the robot travel around a square box forever.

Mechanical limitations will prevent the robot from staying on course forever.
Complete at least one full lap to complete the challenge!

[Click Show hint to add hints here]
Show hint

Previous Next

Copyright©2012 Carnegie Mellon Robotics Academy. All rights reserved.

Loops 3: Looped Movement

This step introduces Loops. The default loop sends Program Flow back to the beginning of the Loop every time it reaches the end (i.e. repeats infinitely).

The Program Flow-based explanation is used because it explains the action of Loops (and later Switches) in a systematic way that avoids common misconceptions.

Mini-Challenge: Square Lap 1

Challenges students to modify the commands inside the loop so the robot makes its way around a square box.

The key to this challenge lies in the symmetry of the box: you can move the same distance for each side, and turn 90 degrees after each movement.



Decisions - Loops

Topics Covered

- Count Mode
- Lesson Links
- Starting Program: MoveLoop.ev3

Check Your Understanding:

- A Loop set to Count Mode will send the Program Flow back...
 - Every time, forever
 - Only a limited number of times
 - If the Touch Sensor is not pressed when the Flow reaches the end of the Loop
 - If there is nothing after the Loop
- What does it mean for a Loop to be "Conditional" in the EV3 Programming Software?
 - It only sends the Flow back under certain conditions
 - The entire Loop can be skipped under certain conditions
 - The Loop runs faster after it is trained, or "conditioned"
 - The code runs every time, no matter what
- What is the "condition" in this Loop based on?
 - The distance the robot has traveled
 - The value of the Touch Sensor
 - The number of times the Loop has sent the Flow back
 - The number of seconds the Loop has been running

Try It!

Try It! 1 | Other Counts
Try changing the number in the Loop's Count setting to 2, then running the program.

What happens?

Mini Challenge

Mini Challenge 1: Square Lap 2
Program your robot to make just one lap around the box.

Modify your Challenge program from the "Looped Movement" (previous lesson) section so that the robot stops after completing exactly one lap around the box.
Do not write each movement by hand – use a Loop!

Loop 4: Loop with Count Control

This step introduces the idea that Loops can end, by using loops that only repeat a certain number of times.

Loops that only repeat sometimes are called Conditional Loops. In this step, “sometimes” is based on the loop count; in the next step, a Sensor will make the decision.

This program builds on the program from the previous step. The “Starting Program” linked to the right of the video provides a working copy of the previous step’s program if needed.

Try It!: Other Counts

Prompts students to use a different number of Counts in the Loop.

Mini-Challenge: Square Lap 2

Instead of looping forever, students are challenged to limit the lap-running behavior to a single full lap (4 sides of the box).



Decisions - Loops

Loops 5 : Loop with Sensor Control

This step continues modifying the program from the previous steps by using a Sensor to decide whether to repeat or not.

There is a common misconception that a “Sensor-Controlled” Loop will continuously monitor the sensor and end the loop immediately when, for example, a threshold is reached.

However, that is not the case; the Sensor is only checked while the repeat-or-end decision is being made at the end of the Loop, not while the commands inside the Loop are running.

This program builds on the program from the previous step. The “Starting Program” linked to the right of the video provides a working copy of the previous step’s program if needed.

Check Your Understanding:

- When the Loop is set to Ultrasonic Sensor as shown below, what “condition” will cause it to pass Program Flow through (instead of sending it back)?

The robot sees something within 30 centimeters immediately after moving forward

The robot sees something within 30 centimeters immediately after backing up

The robot sees something farther than 30 centimeters away

The program repeats itself 30 times

- If you want to make a Loop end based on a sensor reading, you should...

Set the Loop’s mode to “Sensor”

Select the sensor you want while the program is running

Set the Loop’s Mode to Count Sensors

- When does a Sensor Loop check the sensor?

Continuously while inside the Loop

At the beginning and end of the Loop

At the end of the Loop only

At the end of every block within the Loop

Try It!

Try It! 1 Other Sensors

Set the Loop’s mode to Touch Sensor instead of Ultrasonic Sensor. Make sure there is a Touch Sensor on the robot. Run the program and try to get the robot to stop!

What happens?

Mini Challenge

Mini Challenge 1: Square Lap 3

Make your robot run laps around the box until it encounters an obstacle in its way. The obstacle will be placed so that it is visible to the robot immediately after it turns.

Loops 5: Loop with Sensor Control

This step continues modifying the program from the previous steps by using a Sensor to decide whether to repeat or not.

There is a common misconception that a “Sensor-Controlled” Loop will continuously monitor the sensor and end the loop immediately when, for example, a threshold is reached.

However, that is not the case; the Sensor is only checked while the repeat-or-end decision is being made at the end of the Loop, not while the commands inside the Loop are running.

This program builds on the program from the previous step. The “Starting Program” linked to the right of the video provides a working copy of the previous step’s program if needed.

Try It!: Other Sensors

Prompts students to try using the Touch Sensor to end the Loop instead of the Ultrasonic Sensor.

The same rules about the timing of sensor responsiveness still apply.

Mini-Challenge: Square Lap 3

Challenges students to adapt their Square Lap program to end based on an Ultrasonic Sensor value.

The program will only stop for obstacles that are visible to the robot when the Loop ends (typically just after turning).



Decisions - Loops

Loops 6: Loop Review

Sample solutions and explanations for the mini-challenges in this chapter can be found on this page.

Loops 6 : Loop Review

Program Review: Infinite Square Lap

- The program shown below is sample code for mini-challenge in the "Looped Movement" lesson.

Hover on any block in the program to view a detailed explanation.

This Scratch script uses a control loop to repeat a sequence of movement and sound blocks. It includes a loop counter variable named "laps" and a condition to check if the loop has reached a certain number of laps.

Program Review: One Square Lap

- The program shown below is sample code for mini-challenge in the "Loop with Count Control" lesson.
- The code is very similar to the solution for Infinite Square Lap solution above, except that the loop is in **Count Mode**.

Hover on any block in the program to view a detailed explanation.

This Scratch script uses a control loop to repeat a sequence of movement and sound blocks. It includes a loop counter variable named "laps" and a condition to check if the loop has reached a certain number of laps, using count mode.

Program Review: Square Lap with Obstacles

- The program shown below is sample code for mini-challenge in the "Loop with Sensor Control" lesson.
- This code is very similar to the solution for One Square Lap solution above, except that the loop is in **Sensor Mode**.
- This is different from using a Ultrasonic Wait Block:
 - Wait Block holds program flow until a condition is met.
 - Loop with Sensor Control does NOT hold program flow. It only decides whether to send the flow back, or let it out of the loop. Because of this, the loop checks the sensor value just once, immediately after the last movement in the loop.

Hover on any block in the program to view a detailed explanation.

This Scratch script uses a control loop to repeat a sequence of movement and sound blocks. It includes a loop counter variable named "laps" and a condition to check if the loop has reached a certain number of laps, using sensor mode. It also includes a sensor control block to handle obstacles.

[Previous](#) [Next](#)

Copyright©2012 Carnegie Mellon Robotics Academy. All rights reserved.



Decisions - Loops

Loops 7 : Container Handling Challenge

Challenge Review

Container Handling Challenge

CONTAINER HANDLING ATTACHMENTS*

*ACTUAL DESIGN MAY VARY

Building Instructions

- Robot Educator → Driving Base 1/18
- Robot Educator → Medium Motor - Driving... 2/18
- Robot Educator → Ultrasonic Sensor - Driv... 3/18
- Robot Educator → Colour Sensor Down - Dr... 4/18

Where can I find the instructions?

In this Challenge, you will use a Loop to program your robot to move a series of containers into a loading zone. The containers to be loaded are placed at irregular intervals, so you will have to use a sensor to detect each one. The robot should then use its arm to transport the container back into the loading zone – marked with a red outline – and release it there.

1. ROBOT STARTS IN LOADING ZONE
2. PROGRAM TO GRAB AND CARRY BOXES BACK TO LOADING ZONE

USE THE ULTRASONIC SENSOR TO DETECT THE BOXES AND THE COLOR SENSOR TO DETECT THE ZONE WHEN RETURNING

Challenge PDF [container_challenge.pdf]

Container Handling Challenge

This step lays out the details for the Container Handling Challenge. Students should work in their teams to complete the challenge objectives.

As noted near the bottom of the page, the easiest method of solving the problem is to use Forward Until Near to get to the next object, Reverse Until Red to return to the starting area, and a Loop to repeat the behavior for all four containers.

Container objects should be large enough for the Ultrasonic Sensor to detect, but thin enough for the arm to close around. Erasers, toilet paper tubes, or cuttings from cardboard boxes or packing materials can all be good candidates. Be wary of sound-absorbing materials, however, which the Ultrasonic Sensor cannot detect.

Container Handling Challenge PDF

Detailed measurements for the board layout, as well as instructions for setting it up, rule details, and hints for solving the challenge can be found in this printable document.

**Introduction to Programming
LEGO® MINDSTORM® EV3**

CHAPTER 7: Container Handling Challenge

In this Challenge, you will use a Loop to program your robot to move a series of containers into a loading zone. The containers to be loaded are placed at irregular intervals, so you will have to use a sensor to detect each one. The robot should then use its arm to transport the container back into the loading zone – marked with a red outline – and release it there.

Rules and Procedures:

- Use four rectangular objects to represent the four containers. These can be built with LEGO pieces or printed with paper if LEGO is not available. The four objects must be placed in a straight line at the start of each run, with a flat side facing the robot, but their distance from each other should be varied randomly.
- The drop-off area for the containers should be marked with a red outline. If red is not available, any color may be substituted as long as it is different from the color of the table surface.
- A container with no parts hanging outside the loading zone may be removed by hand to make room for the next container.
- The robot must return all four containers reliably to the loading area, regardless of where they were located along the initial line.

Hints:

- Use a Loop to repeat similar portions of the behavior. You should not have to write all four runs separately!
- If the robot needs to perform any actions prior to the loop, simply place them before the Loop in the program.
- If the robot needs to perform any actions after the loop, simply place them after the Loop in the program.
- Use Sensors to help the robot locate both the containers and the loading zone, as the exact distances required will be different each time the robot runs!



Introduction to Robot Decisions - Switches

Decisions > Switches Chapter

Switches and Loops have very different effects on Program Flow: whereas Loops could send Flow back to repeat commands, Switches force the robot to choose between alternative paths in the code. Like Conditional Loops, they typically make this decision based on Sensor feedback.

Switches are seldom seen alone in robotics programming, because robots very rarely make isolated, one-time decisions. It is far more common to put a Switch inside a Loop, to allow repeated decision-making.

Key Concepts: Program Flow, Switches, Repeated Decisions (Switches in Loops)

► **Switches 1: Introduction to the Strawberry Plant Sorter**

Introduces the real-world robot (Strawberry Plant Sorter), and the challenge modeled after it (Strawberry Plant Challenge)

► **Switches 2: Robot Configuration**

Contains building and setup instructions for the rest of the chapter

► **Switches 3: Move If Clear**

Uses a single Ultrasonic Switch to make a decision: move forward if there is no object nearby, or turn to the side if there is

► **Switches 4: Looped Decision**

Use a Loop to repeat the Switch-based behavior, allowing the robot to solve certain simple mazes

► **Switches 5: Review**

Explains sample solutions to mini-challenges from this chapter

► **Strawberry Plant Challenge**

Requires students to use repeated decisions and sensors to program a robot to sort a number of colored objects and move them back to a starting zone

Hints:

- Switches do not repeat unless a Loop makes them repeat (the same way a Loop makes any block repeat)

Decisions - Switches



Switches 1 : Introduction to Strawberry Plant Sorter

Strawberry Plant Sorter 2010

Topics Covered

- Strawberry Plant Sorter
- Challenge overview

Check Your Understanding:

Q 1. A Switch will allow your robot to...

- Write very long programs
- Write multiple programs
- Repeat behaviors within a program
- Make a one-time decision

Next

Copyright©2012 Carnegie Mellon Robotics Academy. All rights reserved.

Switches 1: Introduction

Introduces the Strawberry Plant Sorter, the Strawberry Sorter Challenge, and Switches.

Switches 2 : Robot Config

Ultrasonic Sensor Configuration

The Ultrasonic Sensor is required to complete sections of this chapter.

ULTRASONIC SENSOR ATTACHMENT

ULTRASONIC SENSOR PLACEMENT
PLEASE REFER TO THE BUILDING INSTRUCTIONS CONTAINED IN THE EV3 SOFTWARE

Driving Base 1/18

+ Ultrasonic Sensor - Driv... 7/18

Robot Educator Building Instructions

Robot Educator Building Instructions

Where can I find the instructions?

Previous

Next

Copyright©2012 Carnegie Mellon Robotics Academy. All rights reserved.

Switches 2: Robot Config

The robot for this chapter uses the Ultrasonic Sensor, and will eventually add the Color Sensor in the Challenge.



Decisions - Switches

Switches 3 : Move If Clear

Topics Covered: ● Switches

Move if Clear

Check Your Understanding:

Q 1. The robot will move forward...
 if there is no object in front of the Ultrasonic Sensor when the program starts
 if there is an object in front of the Ultrasonic Sensor when the program starts
 if an object passes in front of the Ultrasonic Sensor at any time
 until an object passes in front of the Ultrasonic Sensor

Q 2. The robot makes its decision about whether to move forward or turn...
 Once, when the Switch is reached in the program
 Once, when the Switch sees an object
 Continuously while the program is running
 The robot always moves, no matter what

Mini Challenge

💡 Mini Challenge 1: Color Sensor Compare Switch
The Switch can use sensors to make its decision as well.
• Attach a Color Sensor to the EV3.

ULTRASONIC AND COLOR SENSOR ATTACHMENTS

ULTRASONIC SENSOR PLACEMENT
KEEP THE ULTRASONIC SENSOR YOU ATTACHED FROM THE PREVIOUS PAGE

COLOR SENSOR PLACEMENT
ADD COLOR SENSOR TO THE CONFIGURATION FROM THE PREVIOUS PAGE

Building Instructions

- Robot Educator → Driving Base
- + Ultrasonic Sensor - Driv... → 7/16
- + Colour Sensor Forward - ... → 5/16

Where can I find the instructions?

• Change the Switch's mode to Color Sensor > Compare > Color.

Brick Buttons → Color Sensor → Color

* Put different parts of the color crate or colored cards or in front of the Color Sensor each time you run the program.

Switches 3: Move If Clear

This step introduces Switches, using the Ultrasonic Sensor to choose between moving straight (if no object is in the way), or turning once (if an object is in the way).

The Program Flow-based explanation is used because it explains the action of Switches in a systematic way that avoids common misconceptions, that also explains their relation to Loops.

Mini-Challenge: Color Sensor Compare Switch

Challenges students to use a Color Sensor to decide whether to turn left or right.

This behavior is directly reusable in the sorting task in the Chapter Challenge.



Decisions - Switches

• Change the Switch's mode to Color Sensor > Compare > Color.

• Put different parts of the color crate or colored cards in front of the Color Sensor each time you run the program.

For this mini-challenge, program the robot to turn to the right if it sees green or brown, and left if it sees any other color. The robot only needs to do this once per run, using the color it sees as soon as the program starts.

[Click Show hint to add hints here]

Show hint

Optional Activities

Mini Challenge

Mini Challenge 2: Color Name Reader

Some switches can have more than two branches

- Try creating a Switch with its Mode set to Color Sensor > Measure > Color.

Then, press the tiny + button (Add Case) on the Switch.

Instead of a checkmark or X symbol, each branch now has a color. When run, the Switch checks the Color Sensor, and runs the branch with the matching color.

If no match is found, the branch with the Default Case dot is run.

Switches 3 (continued)

[Optional] Mini-Challenge: Color Name Reader

Challenges students to investigate the Measure mode of the Color Sensor Switch, which supports multiple branches so the robot can act on several different colors differently, rather than having only two choices.



Decisions - Switches

Switches 4 : Looped Decision

This step places the Switch in a more natural context: inside a Loop.

A single decision often occurs too quickly to be useful (or even noticeable). Repeating the decision gives the robot has the opportunity to perform more meaningful behaviors.

Most robot decision-making takes place in a structure resembling this one.

Check Your Understanding:

Q 1. What happens when you place a Switch inside a Loop?
 The robot runs in a circle until told to stop.
 The robot makes a decision once, then repeats the result many times.
 The robot makes a decision many times, taking whatever action is appropriate each time.
 You cannot place a Switch inside a Loop.

Q 2. When a Switch is placed Inside a Loop...
 Click the + tab near the top of the screen
 The software interprets the Switch+Loop structure as a special construct

Try It!

Maze Runner
The simple behavior you wrote in the video can be used as a very simple maze solver!

Build a maze for the robot, made up of square floor tiles.
If you don't have floor tiles, use 30 cm squares. Make sure you adjust your rotation to go one tile, not 3 rotations.

→ Finish
Start

What happens?

Mini Challenge

Mini Challenge 1: Smarter decisions challenge
Enhance your program by adding the following features:

1. Audio Feedback
 - Make the robot say, "Forward" any time it decides to move forward
 - Make the robot say, "Right" any time it decides to turn right
2. Limited Duration
 - The robot may get stuck. Make the program end after it has performed 10 moves in total.

[Click Show hint to add hints here] **Show hint**

Previous **Next**

Switches 4: Looped Decision

This step places the Switch in a more natural context: inside a Loop.

A single decision often occurs too quickly to be useful (or even noticeable). Repeating the decision gives the robot has the opportunity to perform more meaningful behaviors.

Most robot decision-making takes place in a structure resembling this one.

Try It!: Maze Runner

Prompts students to try running their straight-or-turn looped decision behavior in a maze. The robot will go straight whenever it can, and turn when there is a wall in front of it.

Students will need to adjust the distances traveled in the program to match the “tiles” in the maze.

The walls in this maze should be raised walls (as opposed to marked on the ground). Use textbooks or other free-standing obstacles if possible. If “thin” walls are not available, you can use boxes and push the “top” row back one tile to create a “C”-shaped maze.

Mini-Challenge: Smarter Decisions

Challenges students to add features to different parts of their program.

Students must think about where to add each command, which focuses attention on the role that each part of the program plays.



Decisions - Switches

Switches 5 : Switch Review

Program Review: Move If Clear

- The program shown below is sample code for "Move If Clear" lesson.

Hover on any block in the program to view a detailed explanation.

Program Review: One Square Lap

Program Review: One Square Lap

- The program shown below is sample code for mini-challenge in the "Looped Decision" lesson.

Hover on any block in the program to view a detailed explanation.

Previous Next

Switches 6: Switches Review

Sample solutions and explanations for the mini-challenges in this chapter can be found on this page.



Decisions - Switches

Switches 6 : Strawberry Plant Challenge

Challenge Review

Plant Sorter

COLOR SORTER ATTACHMENT*

*ACTUAL DESIGN MAY VARY

ARM CONTROL PLACEMENT
ATTACH THE ROV'S ARM SO THAT IT CAN GRAB AND PLACE BOXES ON THE SIDES.

[CUSTOM] COLOR SENSOR PLACEMENT
ATTACH THE COLOR SENSOR SO THAT IT CAN REAR THE COLOR OF THE BOXES.

ULTRASONIC SENSOR PLACEMENT
ATTACH THE ULTRASONIC SENSOR SO THAT IT CAN DETECT AND APPROACH A BOX

Building Instructions

Driving Base 1/18 +
Colour Sensor Forward ...
Medium Motor - Driving ... 2/18 + Ultrasonic Sensor - Driv... 2/18

Where can I find the instructions?

In this Challenge, you will use a combination of both Loops and Switches to repeat an inspection process on a total of four plants, represented as boxes. The robot will move to a plant, then sort it to either the robot's right side if it is good, or the robot's left if it is bad. Plant colors are randomized for each run.

USE THE COLOR SENSOR TO DETECT WHICH PLANT (BOXES) ARE GOOD OR BAD. GOOD PLANTS GOES TO THE RIGHT OF THE ROBOT, AND BAD PLANTS ARE SORTED TO THE LEFT.

USE THE ULTRASONIC TO DETECT THE PLANTS AND THE ARMS TO GRAB AND MOVE THEM.

Download Challenge PDF [strawberry_challenge.pdf]

Previous **Complete!**

Copyright © 2012 Carnegie Mellon Robotics Academy. All rights reserved.

Strawberry Plant Challenge

This step lays out the details for the Strawberry Sorter Challenge. Students should work in their teams to complete the challenge objectives.

The robot must sort four plants to the correct side based on each plant's color. The good-or-bad decision should immediately suggest a Switch-based decision, and the fact that there are four objects should suggest a Repeated Decision (Switch in a Loop).

The same objects used in the Loops Chapter's Container Handling Challenge can be used for this Challenge. Add colored paper around or on the sides of each object.

Strawberry Plant Challenge PDF

Detailed measurements for the board layout, as well as instructions for setting it up, rule details, and hints for solving the challenge can be found in this printable document.

**Introduction to Programming
LEGO® MINDSTORM® EV3**

CHAPTER CHALLENGE

CHAPTER 8: Strawberry Plant Challenge

In this Challenge, you will use a combination of both Loops and Switches to repeat an inspection process on a total of four plants, represented as boxes. The robot will move to a plant, then sort it to either the robot's right side if it is good, or the robot's left if it is bad.

Rules and Procedures:

• Use four rectangular objects to represent the plants. These can be built with LEGO parts, or cut from cardboard boxes.
• The objects must be placed in a straight line, with at least 5 centimeters space between them.
• The objects should be a random mix of "good" and "bad" plants, and placed in a random order for each run.
• Green, blue, or brown plants are considered "good". Red, black, or yellow plants are considered "bad".
• The robot must correctly sort all plants in a run to complete the Challenge.

Hints:

- Use a Switch to conduct individual inspections.
- Use a Loop to repeat the inspection process.
- According to the rules, when should the robot stop inspecting?



Introduction to Robot Decisions - Switch-Loops

Decisions > Switch-Loops Chapter

Switch-Loops are actually just Switches inside Loops, the same pattern we called Repeated Decisions in the last chapter. However, if a decision-response cycle is repeated quickly enough, it approaches a “continuous” level of responsiveness.

By analogy, a movie looks like it’s moving smoothly – “continuously” – even though it’s really made of still images, because many frames go by very quickly (24 per second). If a robot updates its motor powers very quickly in response to sensor feedback, the robot looks like it’s responding “continuously”.

Key Concepts: Program Flow, Rapidly Repeated Decisions = “Continuous” Control

- ▶ **Switch-Loops 1: Introduction to the Autonomous Tractor**
Re-introduces the real-world robot (Autonomous Tractor), from the Turning Chapter, and the new challenge modeled after it (Obstacle Orchard Challenge)
- ▶ **Switch-Loops 2: Robot Configuration**
Contains building and setup instructions for the rest of the chapter
- ▶ **Switch-Loops 3: Obstacle Detection Failures**
Shows two “intuitive” attempts to add obstacle detection using existing techniques, both of which fail
- ▶ **Switch-Loops 4: Obstacle Detection**
Uses rapid Repeated Decisions to create a working Obstacle Detection program
- ▶ **Switch-Loops 5: Review**
Explains sample solutions to mini-challenges from this chapter
- ▶ **Obstacle Orchard Challenge**
Requires students to use Obstacle Detection to complete the Orchard Challenge with the added element of obstacles

Hints:

- ▶ Switches inside Loops were already used in the last chapter (as “Repeated Decisions”). The important distinction in this chapter is that “Continuous” behaviors require a rapidly-repeating loop, which the “Discrete” movement responses in the previous chapter did not maintain.



Decisions - Switch-Loops

SwitchLoops 1 : Introduction to Autonomous Tracker

Autonomous Tractor
Obstacle Orchard Challenge

Topics Covered

- Autonomous Tractor
- Challenge overview

Check Your Understanding:

Q 1. What is the difference between this version of the Orchard Challenge and the original version in the Tuning Chapter?
A There are more trees
A The robot must complete the challenge within a certain time limit
A There will be randomly placed obstacles in the robot's path
A There is no difference

Next

Copyright©2012 Carnegie Mellon Robotics Academy. All rights reserved.

SwitchLoops 2 : Robot Config

Ultrasonic Sensor Configuration
The Ultrasonic Sensor is required to complete sections of this chapter.

ULTRASONIC SENSOR ATTACHMENT

ULTRASONIC SENSOR PLACEMENT
+ PLEASE REFER TO THE BUILDING INSTRUCTIONS CONTAINED IN THE EV3 SOFTWARE

Driving Base
Building Instructions 1/18

+
Ultrasonic Sensor - Driv...

Robot Educator
Building Instructions 2/18

Where can I find the instructions?

Previous Next

Copyright©2012 Carnegie Mellon Robotics Academy. All rights reserved.

Switch-Loops 1: Introduction

Re-introduces the Autonomous Tractor, and introduces the Obstacle Orchard Challenge, and Switch-Loops.

Switch-Loops 2: Robot Config

The robot for this chapter uses the Ultrasonic Sensor.



Decisions - Switch-Loops

SwitchLoops 3 : Obstacle Detection Failures

Topics Covered

- Switch Loops

Obstacle Detection Failures

Check Your Understanding:

Q 1. Why doesn't this program work for Obstacle Detection?

- The program detects an object and stops before moving all four rotations
- The program doesn't run because it's waiting to detect something in order to run
- The Move Steering Block holds up the program, preventing it from checking other sensors in the mean time
- All of the above are valid reasons on why the program doesn't work

Q 2. Why doesn't this program work for Obstacle Detection?

- The Wait Blocks prevents the flow from reaching the end of the loop and checking Rotation Sensors
- The program ends immediately when it detects an object, regardless of the Loop
- The Loop makes the robot go backwards
- The first Move Steering block only goes for one rotation, then ends the program

Q 3. Instead of using Waiting approach and long movements, the solution you will learn next will involve:

- Rapid checking of sensors
- Sensor recombination fork
- A new multiple-sensor Wait Block
- A new type of Loop Block

Previous Next

Copyright©2012 Carnegie Mellon Robotics Academy. All rights reserved.

Switch-Loops 3: Failures

This step deliberately walks through two programs that do NOT work as a set up for the program that DOES (in the next step).

Students often have strong preconceptions that the methods shown in this video should work with the right tweaks, when in fact, both methods are fundamentally unworkable.

This video attempts to move students toward greater acceptance of that fact, and openness toward the new approach.



Decisions - Switch-Loops

SwitchLoops 4 : Obstacle Detection

Topics Covered

- Switch Loops
- Obstacle Detection

Obstacle Detection

Check Your Understanding:

- Instead of thinking about the four-rotation Obstacle Detection as one big movement, the video suggests thinking about it as...
- Using a brand new type of Wait Block that allows multiple sensor checking
- Downloading the program into the robot and using built-in options to run parts of the program at certain times
- A long series of tiny movements that add up for four rotations
- All of the above are valid suggestions

- How do repeated decisions allow the robot to watch both sensors at once?
 - The program resets and restarts at the beginning everytime the readings of any sensor changes
 - The robot calculates the path with both sensors at first and runs algorithms for the best way to run the course
 - The robot activates a specific sensor at certain times throughout the course
 - Every operation is fast, and does not block other commands from running
- In the final version of the program, the robot ends up processing the Switch inside the Loop...
 - Only once, ever, because it's a Switch
 - Four times because the Loop goes for four rotations
 - Thousands of times, because the robot processes the loop very quickly and encounters the Switch many times

Mini Challenge

Mini Challenge 1: Obstacle Detecting Move Until Black Line

Modify the Obstacle Detection program you wrote so that it will move safely (stopping when an obstacle is in front of it, moving when there is none) until the Color Sensor detects a black line on the table, rather than until the robot has traveled a certain number of rotations.

- Attach a Color Sensor to the EV3.

ULTRASONIC AND COLOR ATTACHMENT*

*ACTUAL DESIGN MAY VARY

Driving Base

Building Instructions

Where can I find the instructions?

Switch-Loop 4: Obstacle Detection

This step introduces the idea that large behaviors can be built up from repeated small behaviors, and that a Switch inside a Loop can create this effect in code.

The rapid repetition of the Switch and Loop decisions allows multiple sensors to be checked, as neither one “blocks” the other by holding up the program flow.

Since the behavior relies on the Loop to continue running, exiting the Loop ends the behavior. Consequently, the sensor condition on the Loop (Motor Rotations > 4 in this case) determines when the behavior ends. The sensor in the Switch determines what the robot does in each given instant until then.

This, of course, only works so long as the program is able to loop rapidly; it would not be able to check the Loop end condition if another block were holding up program flow!

Mini-Challenge: Obstacle Detecting Move Until Black Line

Challenges students to add a Color Sensor to the EV3 and modify the program so that it ends when the robot reaches a black line.

Since the sensor on the Loop determines when the behavior ends, setting the Loop condition to Color Sensor > Color > Black means it will end when the robot reaches the black line.



Decisions - Switch-Loops

The image shows a Scratch project titled "SwitchLoops 5 : Review". The stage has a light blue background with a red border. A green cat sprite is on the stage. The script area contains the following script:

```
when green flag clicked
  [Obstacle Detection Until Black Line v1.0]
  end
```

The script for "Obstacle Detection Until Black Line v1.0" is:

```
repeat [ ]
  if [black line v] then
    [ ]
  end
end
```

Below the stage are "prev" and "next" buttons, and an "Exit" button. At the bottom are "Previous" and "Next" navigation buttons.

Switch-Loop 5: Review

A sample solution and explanation for the mini-challenge in this chapter can be found on this page.



Decisions - Switch-Loops

SwitchLoops 6 : Challenge

Challenge Review

Obstacle Orchard Challenge

ULTRASONIC SENSOR ATTACHMENT

In this Challenge, you will program your EV3 robot to move from its starting area through three rows of fruit trees. The robot must pass along both sides of each row during its run. In addition, however, there will be one or more obstacles placed at random throughout the orchard. The robot should not touch these obstacles; instead, when it encounters one, it should stop moving until they are removed by hand... at which point the robot should continue on its way.

THE STARTING AREA CAN BE LOCATED TO WHEREVER SPACE IS AVAILABLE. OBSTACLES ARE PLACED THROUGHOUT THE ORCHARD AT RANDOM LOCATIONS.

Challenge PDF [obstacle_challenge.pdf]

Download

Previous

Complete!

Copyright©2012 Carnegie Mellon Robotics Academy. All rights reserved.

Obstacle Orchard Challenge

This step lays out the details for the Obstacle Orchard Challenge. Students should work in their teams to complete the challenge objectives.

This Challenge board is set up identically to the Orchard Challenge from the Turning chapter, but contains obstacles that can be randomly placed. The obstacles can be the same as the ones used in the Container Handling or Strawberry Sorter Challenges.

Obstacle Orchard Challenge PDF

Detailed measurements for the board layout, as well as instructions for setting it up, rule details, and hints for solving the challenge can be found in this printable document.

**Introduction to Programming
LEGO® MINDSTORM® EV3**

CHAPTER CHALLENGE

CHAPTER 9: Obstacle Orchard Challenge

In this challenge, you will program your EV3 robot to move from its starting area through three rows of fruit trees. In addition, however, there will be one or more obstacles placed at random locations throughout the orchard. The robot should not touch these obstacles; instead, when it encounters one, it should stop moving until they are removed by hand... at which point the robot should continue on its way.

Rules and Procedures:

- This challenge uses the same game board layout as the Orchard Challenge from Chapter 2 (Turning).
- Like the previous challenge, the robot can start anywhere there is space available.
- Place one to two obstacles randomly alongside a side of a row for the robot to encounter.
- Be aware to not place an obstacle where the robot may bump into when turning a corner.
- When the robot encounters an obstacle, it should stop and wait for the obstacle to be removed by hand. It should then continue moving without additional human intervention.

Starting Area:

RIVER

Hints:

- Use a meter stick or ruler to measure the distances to each line on the board so you know how far you need to move each time.
- The obstacle can be completely removed from the challenge after the robot approaches it and stops.
- Use lower speeds to minimize effects of momentum when turning.



Introduction to Robot Decisions - Line Follower

Decisions > Line Follower Mini-Chapter

Note: The terms “Line Following” and “Line Tracking” are used interchangeably in this chapter.

Line Following is actually just another application of the rapid repeated decisions pattern introduced in Switch-Loops. Consequently, the Line Follower chapter actually contains no new blocks or concepts – it is a new application of an existing idea.

As it is more practice and less new material, the Line Follower chapter is condensed into two pages and two videos.

Key Concepts: Applying “Continuous Control” in a new situation, Line Tracking

► **Line Tracking 1: Introduction to the AMTS**

Introduces the real-world robot (Automated Material Transport System [AMTS]) and the new challenge modeled after it (Line Tracking Challenge). Also contains the building and setup instructions for this chapter.

► **Line Tracking 2: Line Tracking**

Explains Line Tracking as a use of Repeated Decisions and walks through a basic program that performs Line Tracking. Also lays out the Line Tracking Challenge, which requires students to move a crate along a curved path.

Hints:

- Line Following is a useful option any time there are suitable floor markings, including many robot competitions boards.
- Just as with Obstacle Detection, the Line Following behavior relies on the Loop’s condition to know when to stop, and the Switch’s condition to decide what motor commands to issue in any given instant.



Decisions - Line Follower

Line Follower 1 : Introduction to Automated Material Transport System (AMTS)

Automated Material Transport System
Line Tracking Challenge

Topics Covered

- Automated Material Transport System (AMTS)
- Challenge overview

Check Your Understanding:

Q 1. What environmental feature will the robot use to help it navigate in this challenge?

- A line on the ground
- IR beacons on walls
- Volume levels of environments
- It is facing Northward

Q 2. What programming concept will be used to create this behavior?

- Discrete behaviors
- Multitasking
- Repeated decisions
- Bad decisions

Robot Configuration

Color Sensor (Facing Down) Configuration

The Color Sensor is required to complete sections of this chapter.

COLOR SENSOR (DOWN) ATTACHMENT

COLOR SENSOR (DOWN) PLACEMENT

MODIFY THE COLOR SENSOR SO THAT IT POINTS DOWN, IN ORDER TO DETECT LINES ON SURFACES

Driving Base 1/16

Colour Sensor Down - Dr... 4/18

Where can I find the instructions?

Next

Copyright 2012 Carnegie Mellon Robotics Academy. All rights reserved.

Line Tracking 1: Introduction & Config

Introduces the Automated Material Transport System (AMTS), the Line Tracking Challenge, and the Line Following/Tracking behavior.

This page also includes the building instructions for this chapter.



Decisions - Line Follower

Line Tracking 2: Challenge

This step explains Line Tracking, which uses the same Rapidly Repeated Decisions logic used in Obstacle Detection.

The Challenge for this chapter is found farther down this page.

Check Your Understanding:

1. If the robot is seeing Black, what can it tell about its position?
 - It is over the line
 - It is outside the line to the left
 - It is outside the line to the right
 - It is facing Northward
2. Suppose the robot is tracking the left edge of the line, and sees Black. Which way should it move?
 - Forward and to the left, because the left edge is that way
 - Forward and to the right, because the left edge is that way
 - Straight forward because the line will curve eventually
 - Turn left in place, because the left edge is that way
3. Which basic program pattern is being used in this Line Tracking behavior?
 - Sequential Movements
 - Parallelism
 - Repeated Decisions
 - Variables

Mini Challenge

Mini Challenge: Track Line for Rotations
Modify the behavior so that it stops line tracking after 4 motor rotations on Motor B

PROGRAM THE ROBOT TO STOP LINE TRACKING AFTER MOVING FOR FOUR ROTATIONS

(Click Show hint to add hints here) [Show hint](#)

Line Tracking 2: Challenge

This step explains Line Tracking, which uses the same Rapidly Repeated Decisions logic used in Obstacle Detection.

The Challenge for this chapter is found farther down this page.

Mini-Challenge: Line Track for Rotations

Challenges students to make the Line Tracking behavior end after 4 rotations. The same logic applies here that was used to make Obstacle Detection end after 4 rotations.



Decisions - Line Follower

Mini Challenge: Track Line for Rotations
Modify the behavior so that it stops line tracking after 4 motor rotations on Motor B.

PROGRAM THE ROBOT TO STOP LINE TRACKING AFTER MOVING FOR FOUR ROTATIONS.

[Click Show hint to add hints here.] [Show hint](#)

CHALLENGE

LINE TRACKING ATTACHMENT*
*ACTUAL DESIGN MAY VARY.

ARM CONTROL PLACEMENT
• ATTACH THE ARM IN ORDER TO GRAB ONTO THE LEGO CRATES OR CUSTOM BOXES

COLOR SENSOR (DOWN) PLACEMENT
• ATTACH THE COLOR SENSOR SO THAT IT POINTS DOWN, IN ORDER TO DETECT THE TRACK. NOTE: THE COLOR SENSOR MAY NEED TO BE ELIMINATED SLIGHTLY FROM THE GROUND TO WORK PROPERLY

Robot Educator → Building Instructions → Driving Base 1/18

+ Medium Motor - Driving ... 2/18 + Colour Sensor Down - Dr... 4/16

Where can I find the instructions?

In this challenge, you will program your EV3 robot to grab a cargo crate from the pickup spot, follow the line track and drop the crate off in the drop-off zone.

1. PICK UP THE COLORED CRATE FROM PICKUP ZONE
2. FOLLOW THE LINE AND DROP THE CRATE INTO THE DROPOFF ZONE

Challenge PDF [lineTrack_challenge.pdf] [Download](#)

Previous Complete

Line Follower 3 (continued)

Challenge: Line Tracking Challenge

Challenges students to have the robot pick up a crate and follow a complex line using a series of line-tracking behaviors.

Students can greatly improve the performance of the robot by modifying motor power levels.

The ratio of the left and right motor powers determines the “sharpness” of the robot’s motion (e.g. 75/25 follows the same course as 30/10).

High motor powers will make the robot move quickly to save time, while lower ones will make it move more slowly (but stay on the line better).

Best results are achieved by using multiple line tracking behaviors back-to-back, with each one having a different combination of motor powers. Each can run for a set distance or length of time, then end, allowing the next to run.

Line Tracking Challenge PDF

Detailed measurements for the board layout, as well as instructions for setting it up, rule details, and hints for solving the challenge can be found in this printable document.

**Introduction to Programming
LEGO® MINDSTORM® EV3**

CHAPTER 10: Line Track Challenge

In this challenge, you will program your EV3 robot to grab a cargo crate from the pickup spot, follow the line track and drop the crate off in the drop-off zone.

Rules and Procedures:

- Build the course shown above using black electrical tape on a light surface.
- The robot must pick up the box using its arm, then follow the line to the drop-off point, and finally drop the box.
- Program your robot to deliver the box in the shortest time possible!

Hints:

- You can use the Loop Mode setting to advise when a Line Track behavior ends. This program will then run whatever comes next, even another Line Track.
- Use multiple line tracks with different “sharpness,” one after another to handle parts of the board.
- Use the Line Track behavior with the Line Sensor and the Line Sensor with the Line Sensor using the Crossing Order or the Move steering blocks.
- You may find it advantageous to track the right side of the line in some places. Which way should the robot go when it sees black to get to the right edge of the line? Which way should it go to get back after it drives off?



Final Challenge Resources

Resources 1 : Flowcharts

The Flowcharts video .explains how a robot makes decisions. This video is designed to introduce students to flowcharts.

Key Concepts: Problem Solving: Breaking Down large problems

Resource 2 : Iterative Design

The Iterative Design video gives students a process that they can use to solve their programming problems.

Key Concepts: Building Up a solution with Iterative Design

Oftentimes, students' first inclination when solving a problem that involves programming is to simply sit down and start coding. This is generally counterproductive if the solution is not immediately obvious – students are more likely to get lost than succeed if they do not "get their bearings" and pick a sensible direction first. [NGSS: MS-ETS1, HS-ETS1-2]

Resources 3 : Project Planning

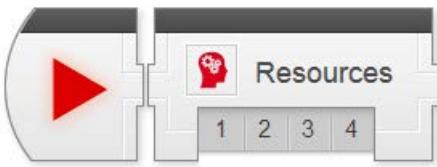
The Project Planning video .explains how important planning is when working in teams.

Key Concepts: Problem Solving: Planning, Breaking Down large problems

Resources 4 : Engineering Process

The Engineering Process video .describes a logical and systematic way to solve engineering projects.

Key Concepts: Problem Solving: Planning, Breaking Down large problems



Final Challenge Resources

Resources 1 : Flowcharts

Topics Covered: Flowcharts

Check Your Understanding:

Q 1. What is a Flowchart?
A graphical representation of a robot's plan of action, including decisions
A series of pipes and wires that illustrate electricity flow
The overall map of a program's progress
The document that tracks the number of weeks left in the project cycle

Q 2. Why are Flowcharts important?
The flowchart calculates the trajectory of the rejected plants
Robots internally reprocess all instructions into flowcharts in order to think
EVA programs can be written in Flowcharts and loaded directly into the robot
They help programmers visualize the decision-making process on the robot

Next

Copyright©2012 Carnegie Mellon Robotics Academy. All rights reserved.

Resources 2 : Iterative Design

Topics Covered: Iterative Design

Check Your Understanding:

Q 1. What method does this video recommend for building a solution to a problem?
Build almost everything at once, then try to get the last part to fit
Solve a part, add it to the solution, test the combined whole, then repeat
Solve all the parts separately, then combine them in one step
Construct the entire solution all at once

Previous Next

Copyright©2012 Carnegie Mellon Robotics Academy. All rights reserved.

Final Challenge Resources 1

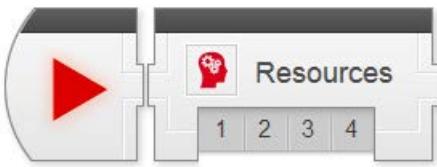
Flowcharts Video

The Flowcharts video .explains how a robot makes decisions. This video is designed to introduce students to flowcharts

Final Challenge Resources 2

Iterative Design Video

The Iterative Design video gives students a process that they can use to solve their programming problems.



Final Challenge Resources

Resources 3 : Project Planning

Topics Covered

Project Planning

Check Your Understanding:

Q 1. What main topic does this video address?

- How to build a car out of LEGO bricks
- How to write the most efficient program code
- How a robot "thinks" about its surroundings
- How to coordinate a team of people working together on the same problem

Q 2. What does a Design Specification do?

- Explain the importance of the problem being solved
- Align team members' ideas of what is being built
- Distribute the work evenly among team members
- Keep track of the days that people have shown up to work on the robot

Resources 4 : Engineering Process

Topics Covered

Engineering Process

Check Your Understanding:

Q 1. What main topic does this video address?

- How to coordinate a team of people working together on the same problem
- What engineering is, and how it works
- Imagining the solution to a problem
- Types of terrain that are most suitable for robot use

Q 2. Which of the following is not an important element in a good Engineering Process?

- Researching the problem
- Planning the development
- Prototyping the solution
- Testing the prototype
- Commercializing the product
- Ignoring resource limitations

Previous

Complete!

Final Challenge Resources 3

Project Planning Video

The Project Planning video .explains how important planning is; especially when working in teams.

Final Challenge Resources 4

Engineering Process Video

The Engineering Process video .describes a logical and systematic way to solve engineering projects.



Search and Rescue Challenge

Final Challenge : Search and Rescue Challenge

Challenge Overview

Search and Rescue Robots

Final Challenge

2:28 4:15

FINAL CHALLENGE ATTACHMENT*

*ACTUAL DESIGN MAY VARY

ARM CONTROL PLACEMENT
ATTACH THE ARM IN ORDER TO GRAB ONTO THE LEGO CRATES OR CUSTOM BOXES

ULTRASONIC SENSOR PLACEMENT
MAKE SURE THE ULTRASONIC SENSOR HAS A CLEAR VIEW FORWARD TO DETECT CONTAINERS

COLOR SENSOR (DOWN) PLACEMENT
ATTACH THE COLOR SENSOR SO THAT IT PENTS DOWN, IN ORDER TO DETECT LOADING ZONE

Robot Educator

Building Instructions

Driving Base 1/16

Medium Motor - Driving ... 2/16

Ultrasonic Sensor - Driv... 7/16

Color Sensor Down - Dr... 4/16

Where can I find the instructions?

Building Instructions in EV3 Education Edition

2:28 4:15

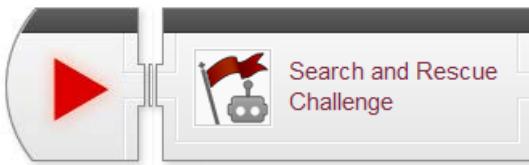
Search & Rescue Challenge

This step lays out the details for the final Search and Rescue Challenge. Students should work in their teams to complete the challenge objectives.

This challenge unfolds in two stages, to help students focus on breaking down the problem and building up the solution.

Building Instructions

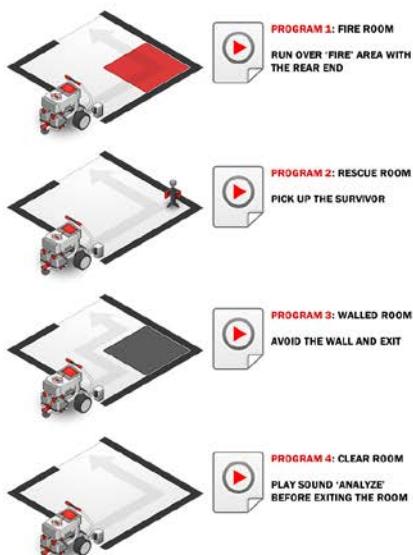
Students should be allowed to make minor modifications to their robots during this Challenge, although they should not be necessary; all tasks can be completed with the standard attachments as shown.



Search and Rescue Challenge

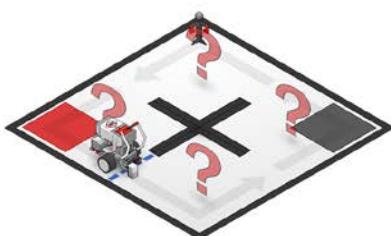
PHASE 1

1. WRITE 4 SEPARATE PROGRAMS FOR EACH ROOMS
2. ROBOT CAN ENTER EITHER ENTRANCE OF THE ROOM
3. ROBOT MUST EXIT THE ROOM AFTER COMPLETING THE OBJECTIVE OF EACH ROOM



PHASE 2

1. WRITE 1 PROGRAM THAT WILL TRAVEL ALL 4 ROOMS
2. THE ROBOT MAY START AT ANY ROOM'S ENTRANCE
3. THE ROBOT'S TRIP CAN BE EITHER CLOCKWISE OR COUNTER-CLOCKWISE
4. THE ROBOT MUST RETURN TO WHERE IT STARTED
5. THE LOCATION OF THE ROOMS WILL BE RANDOMIZED EACH RUN



Challenge PDF
[Final_challenge.pdf]

Previous

Complete

Copyright © 2012 Carnegie Mellon Robotics Academy. All rights reserved.

Search and Rescue Challenge (cont'd)

Phase 1

In Phase 1, students should focus on completing the tasks found in each room separately, using a separate program for each room.

In Phase 2, these individual behaviors will be combined with additional logic that makes the correct behavior run at the correct time, and repeats the process to cover four rooms.

In the “iterative” model, this is the first step – the foundation upon which the next steps will be built.

Phase 2

In Phase 2, students work on the logic that decides which of the four room-specific behaviors should run, and at what times. It will then rely on the existing, tested behaviors from Phase 1 (with small modifications as necessary) to complete the room task.

In the “iterative” model, this step builds upon the first step’s behaviors, and instead focuses on choosing the right one at the right time.

Search and Rescue Challenge PDF

Detailed measurements for the board layout, as well as instructions for setting it up, rule details, and hints for solving the challenge can be found in this printable document.

The screenshot shows the software interface for the LEGO MINDSTORM EV3. At the top, there's a navigation bar with a play button, a chapter title 'CHAPTER 10: Line Track Challenge', and a 'CHAPTER CHALLENGE' button. Below the title, there's a diagram of a line track with a robot. A box labeled 'Rules and Procedures' contains the following text:

In this challenge, you will program your EV3 robot to grab a cargo crate from the pickup spot, follow the line track and drop the crate off in the drop-off zone.

Rules and Procedures:

- Build the cargo crates using Block.
- The robot must start on a light surface.
- The robot must pick up the box using its arm, then follow the line to the drop-off point, and drop the box.
- Program your robot to deliver the box in the shortest time possible!

Below the rules, there's a 'Hints' section with several bullet points:

- You can use the Loop Mode setting to adjust when a Line Track behavior ends. The program will then run an If-Then loop, even after the Line Track.
- Use the Angle block to make the robot turn in a straight line.
- Adjust the “sharpness” of the robot’s motion using the Steering slider on the Move Steering Block.
- You may find it advantageous to track the right side of the line in some places. When may should the robot turn?

Reproducibles

Table of Contents

1 Unit Worksheets

- 93 Movement Straight Question and Answer Key
- 95 Movement Turning Question and Answer Key
- 97 Sensors Touch Question and Answer Key
- 99 Ultrasonic Sensor Question and Answer Key
- 101 Gyro Sensor Question and Answer Key
- 103 Color Sensor Question and Answer Key
- 105 Decisions Loops Question and Answer Key
- 107 Decisions Switches Question and Answer Key
- 109 Decisions Switch Loops Question and Answer Key
- 113 Decisions Line Follower Question and Answer Key
- 115 Robot Engineering Question and Answer Key

117 Robot Programming Worksheets

- 117 What are Behaviors?
- 118 What are Flowcharts?
- 119 What is Pseudocode?
- 120 What is a Robot?

121 Engineering Handouts

- 121 Engineering Process Handout
- 122 Engineering Process Steps
- 123 Using Gantt Charts
- 124 Using PERT Charts
- 125 Introduction to the Engineering Journal
- 126 Using the Engineering Journal

128 Rubrics

- 128 Student Work Habit Evaluation Rubric
- 129 Example Writing Rubric
- 130 Student Presentation Rubric
- 131 Design Review Rubric
- 132 Proposal Writing Rubric

133 Addendum: Bonus Materials

- 133 Data Logging Investigation
- 138 Data Wires and Logic

