Project Ataris

Introductions

Kyely
Systems Engineer

Jason
Software Engineer

Anna Sophia
Systems Engineer

Travis
Systems Engineer

Grace
Project Engineer

Brennan
Systems Engineer

Mission

The Ataris Quest: Operation Pathfinder

In the year 2085, humanity has successfully landed on the first permanent research station on a newfound planet, Ataris. A group of engineers have been selected to remotely assist in one of the most critical missions to date: guiding an advanced robot explorer, codenamed Pathfinder, across the treacherous, unknown terrain. Pathfinder must explore a newly discovered terrain rich with potential resources but filled with obstacles and variable environmental conditions. The robot's ability to navigate, analyze and make decisions autonomously will be key to the success of the mission.

Mission Goals

Map the Terrain

Scientists from the research station have marked out a network of paths that the robot must traverse and complete various goals

Obstacle Avoidance

The robot shall autonomously alter its course to avoid collisions upon detecting obstacles in its path.

Sound Detection

The robot shall increase its movement speed when a sound is detected.

Light Finder

The robot shall be able to detect and follow the direction of the light, adjusting its path to navigate towards the light source.

Line Following

The robot shall be able to detect and follow a drawn path, adjusting its movement to stay on the path until it reaches its destination.

Class Objectives

*General Atomics team will be present

11/6-11/7 – Creating Teams

Students should get into teams and establish team names. Assign roles on who will work on building the robot, who will be building the landscape, and who will be coding. Can unbox robot kit to look through the supplies, should start building and explore the software platform on the website. Charge up battery pack for robot.

11/15 – Mission Kickoff*

Students will meet the General Atomics Team, obtain mission requirements, and go over basics of programming language on the Makeblock development platform. This is also an opportunity to ask any initial questions of the project they may have.

11/20 – Working Session

Students will meet with their teams to build the robot and start coding to their requirements. Brainstorm on how they might be able to present their demonstration (for ex. Are you going to need to make a track for your robot? Are you going to have to create landmarks for your robot to demonstrate functionality?)

11/22– Requirements*

At this point, students should have their robot 100% built up and code almost done (if not completely finished). We will have a mini lesson on requirements and how to verify requirements. The remainder of the time will be to help answer any questions or help any team with their code development. If any teams are completely done, what other functions can you program your robot to do?

12/6– Demo Day 1*

On this day, students will be demonstrating their robot and programs to the General Atomics team. Reflection on what went well and what didn't go well. Showcase how well they were able to meet their requirements. For the sake of time, each team will get 3 minutes.

12/13– Demo Day 2 / Q&A *

Finishing up Demos if needed. Rest of the time will be for Q&A with the General Atomics team

Project Rubric

Extra points may be awarded for the bonus tasks

Score	1	2	3	4	5
Task Performance/Functionality	Poor build. Robot was built incorrectly and cannot stand on its own or perform as intended. Does not meet requirements	Adequate build with noticeable weaknesses. Robot works but structurally unstable. Potentially incorrect configuration was chosen. Barely meets requirements.	Robot is functional but there may be minor structural or functionality issues. Meets about half of the requirements	Robot is well-built, creative, and mostly meets the challenge requirements	Exceptional build that is both highly creative and functional. The robot demonstrates sturdiness and meets all the challenge requirements
Programming	Poor programming, resulting in little to no task completion or robot control	Basic programming with frequent errors. Robot completes tasks but requires constant adjustment or manual intervention	Satisfactory programming with some inefficiencies. The robot performs most tasks, though there may be frequent errors or inconsistencies.	Good programming with minor issues. Robot performs tasks with occasional errors but works reliably most of the time.	Programming is highly efficient, logical, and well organized. Robot performs tasks autonomously with accuracy and consistency
Creativity/Visual Appeal	Built to standard manual instructions.	Added 1 additional item to standard manual that does not add to performance or functionality	Added a couple items to the standard but does not contribute to the performance or functionality. Some level of effort for increasing visual appeal	Added several items to the standard that enhance the ability to meet requirements. Visible effort for increasing visual appeal	Added several items to the standard to enhance ability to meet requirements and showcases team's individual strengths through demonstration
Demonstration and Presentation	Poor or missing presentation and documentation. The team is unable to clearly explain their work or process	Basic presentation with limited documentation or explanation. The process is explained superficially or missing key details.	Satisfactory presentation and documentation. The team explains their work but lacks depth or detail.	Good presentation and documentation with minor gaps. The team explains their process clearly but could improve in detail or reflection	Team provides an exceptional presentation and documentation. They clearly explain their design, programming, and problem-solving processes with thorough detail and reflection
Teamwork and Collaboration	Poor teamwork with little to no collaboration. One member handled most of the work.	Weak teamwork, with only a few team members actively contribution. Communication and collaboration are lacking	Satisfactory teamwork with occasional communication or collaboration issues. Contributions may be uneven	Good teamwork with minor imbalances. Most members contribute and collaborate effectively	Excellent teamwork with all members contributing equally. Team demonstrates strong collaboration, communication, and problem solving skills

Programming Language

Programming is like giving a set of instructions to a computer to make it do something. A computer will follow given instructions (called code) step by step. Program the Arduino board to meet the provided requirements.

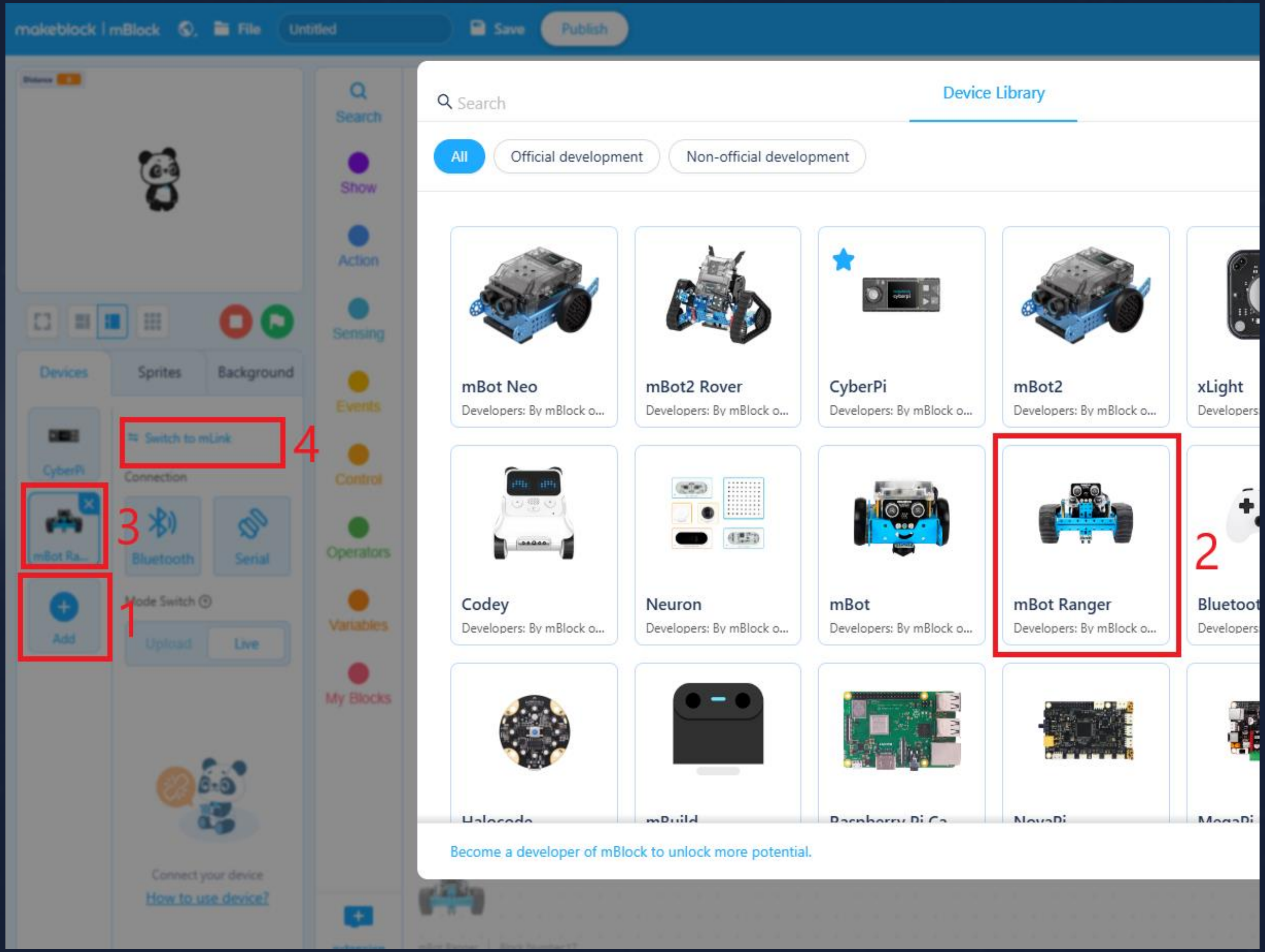
We will be using Block as our programming language. Put together a set of blocks to make the mbot Ranger perform the correct actions!

Environment Setup

Use Chrome browser
https://mblock.cc/ → 'Code with Block'

Connect to your robot:

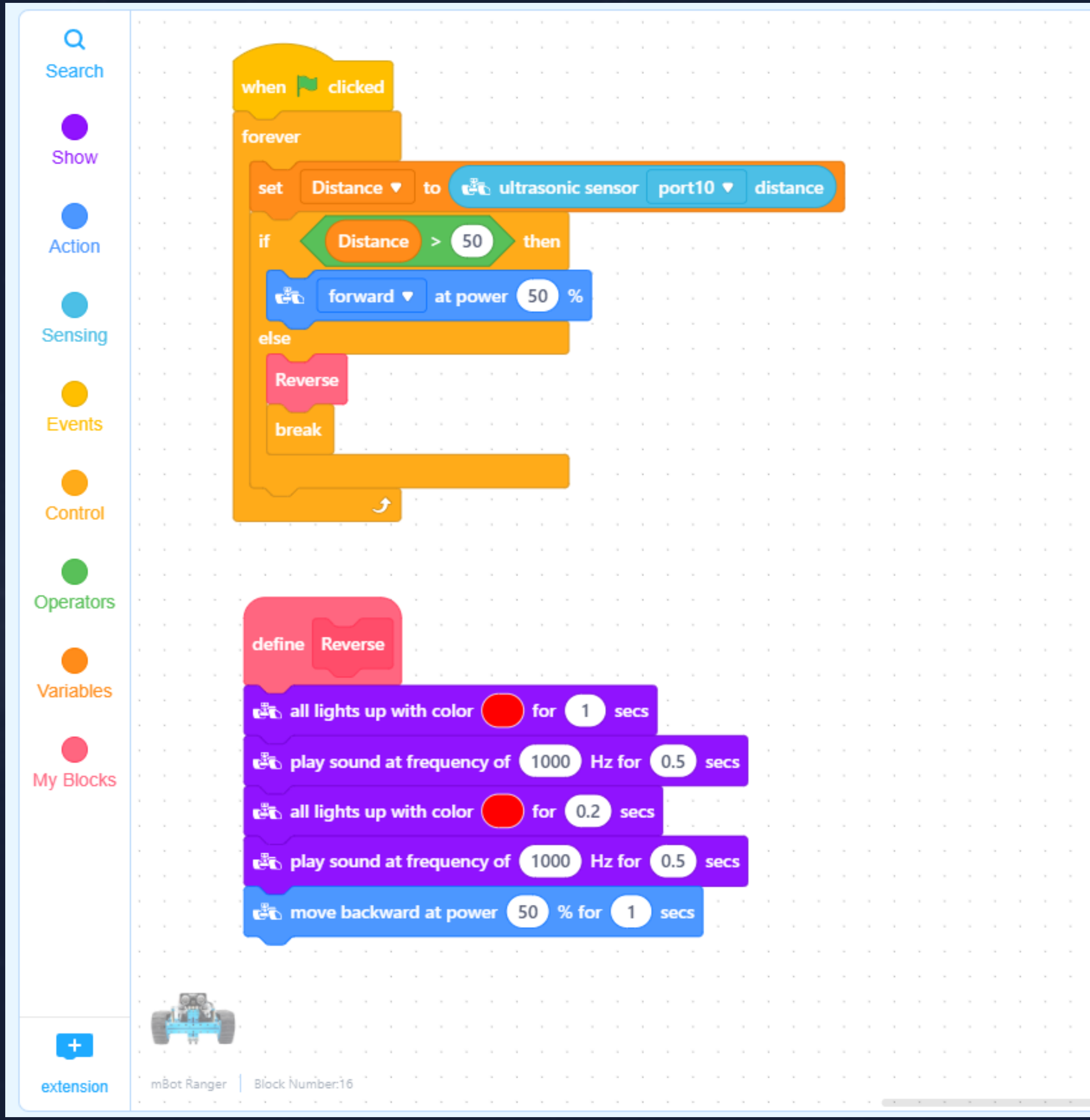
1. Add new Device
2. Choose mBot Ranger
3. Select the chosen device on the left
4. 'Switch to direct connection'
5. Select Serial to connect via USB cable
6. Wait until you hear the robot beep
7. Disconnect and select Bluetooth to connect



Navigating the IDE

- Drag and drop blocks onto the scratch pad
- Piece them together like puzzles; the shapes must match for valid block of code
- Drag and drop blocks to the left to delete
- Double click block of code to execute code
- Code is executed line by line and only one item can be performed at a time

Obstacle Avoidance Example



Requirements Lesson

What are Requirements?
Requirements are like a list of goals for a project or task. The requirements would describe what the result looks like and what it should be able to do.

Type of Requirements
Functional
Non-functional

How are Requirements determined?
Requirements may be determined by understanding the mission goal and talking to the customer.
What's needed to make something work well?
What do people want or need?

Why are Requirements Important?
Requirements help us make sure things turn out the way we want. When we know exactly what's needed, we can plan better and avoid problems

Increasing Scope: More You Can Do (Pt. 1)

Signaling Success

Think about how you can verify that your robot is doing what it's supposed to. Does it light up after finishing a task? Does it play a jingle? Get creative – design and implement an intuitive signal that shows when your project succeeds at different tasks!

Dealing with Failures

Conduct a Failure Modes and Effects Analysis (FMEA)! Determine the ways your robot can fail, why it this could happen, and how you resolve those issues! Make a chart showing the results of this analysis. Remember, turning it off and on again is a valid solution!

Looking to the Future

With everything you learned in this project, what would you do to make your design better? What real world applications does your work have? What do you want to learn next?

Increasing Scope: More You Can Do (Pt. 2)

Logo Design

Design a logo and come up with a clever name for your project! A good name is relevant to the goal, memorable, and easy to remember, and a logo should be easily readable and relevant to the name and project.

Elevator Pitch – Sell it!

An important part of any project is how you present it! Work on a short pitch that you can use to help people understand what your project does, how it works, and why they should be interested!

Reflections

What did you enjoy about the project?

There are elements of coursework that span across all disciplines of engineering such as geometry, calculus, physics, chemistry, digital electronics, computer science, etc. We would encourage you to continue joining clubs or activities that tie in real world engineering applications to home in on your interests. Remember that even if you don't use something from the classroom in the future, the value is in learning the ability to problem solve, think outside the box, work as a team, and drive a task to completion!

Building the Robot

If you enjoyed more of the hands-on aspect of building the robot and testing it out, consider looking into mechanical engineering or electrical engineering disciplines and their real-world applications

Coding The Robot

If you enjoyed more of the coding aspects and troubleshooting where the code might not have been working or testing how much you can program the robot to do, consider looking into computer science related disciplines. Computer science has many applicable fields and industries.

Overall Project

If you enjoyed the project as a whole, but don't fall under either software or hardware interests, look into systems engineering. Systems engineering allows you to view the project as a whole, ensuring all the components will work together as expected.

Reflections

What did you learn?

- Problem Solving and Design Thinking – By brainstorming, designing, building, and testing your robot, this mimics the engineering design process used in real-world applications
- Programming and Automation – By coding your robots, you explored the basics of automation and robotics which are foundational in industries like engineering, healthcare, and beyond!
- Mechanical Engineering Concepts – Building robots allowed you to explore principles like structural stability and improvement. You had to make sure your robot was built correctly to support the tasks you wanted it to do.
- Electrical Systems – Working with sensors, motors, and circuits allowed you to look at how the electrical system should work together
- Teamwork and Collaboration – Engineering projects often require effective communication and teamwork. We do this every day in our work environment where we collaborate to solve problems. You had to work not only with your teams but with other teams to overcome any challenges.
- Iterative Testing and Prototyping – Testing your robot and refining your build, code, or design teaches the importance of iteration, a critical skill for optimizing performance and reliability. Build, test, repeat!

Thank you for having us!

Q&A