Cyber Robot Challenge

Level: High School
Type of Contest: Team
Composition of Team: 3–4 students per team
Number of Teams: One entry per school
****Next Gen Science Standard(s): HS-ETS1-2. , HS-ETS1-4.**[1]

NOTES

Submission of all program files for competition must be received electronically (via "Dropbox") **no later than 10 calendar days before the competition**.

No submissions will be accepted 3 days after the cutoff date (7 calendar days before competition).

There will be a flat 10-point penalty assessed for late submissions.

Programs will be installed on the computers and will be available to students on the day of competition.

For this event, no removable media (CDs, USB thumb drives, etc.) will be allowed on the day of competition. Additions or modifications to programs on the day of competition must be entered manually by the students.

Python Software can be downloaded from the Python website at:

http://www.python.org/download/releases/2.5.4/.

Overview

Student teams will use the Python programming language to program a **virtual robot** to navigate through a series of **computer networks** executing a mission. In this year's mission, the robot must search for and disable viruses that have infected each network while simultaneously finding and collecting **packets** in a particular order and observing **clues** to help in cracking the network's **secret passphrase**. Each network is represented as a maze that the robot must explore, and some of the viruses can only be defeated by solving **cryptographic puzzles**.

[1] NGSS Lead States.(2013). *Next Generation Science Standards: For States, By States*. Washington, DC: The National Academies Press: http://www.nextgenscience.org/next-generation-science-standards.

For each network, the robot must find its way through the maze **autonomously**; that is, once the program is started, no user input is accepted to navigate the robot. However, the networks may be reattempted multiple times, and the robot logic can be altered between attempts.

Each team's performance will be measured on the basis of the number of viruses defeated in the time provided during the challenge, with extra points being awarded for collecting all the network packets in order and cracking the networks' secret passphrases. Detailed scoring criteria are discussed in the section entitled Detailed Specifications and Judging Guidelines

The Networks

Each network is represented as a maze of walls aligned to a grid of square cells, similar to a chessboard without the squares shown. A network may assume any shape and be of any size and is not limited to being a simple square or rectangle. The robot will always begin in the center of an arbitrary cell in the network.

Figure 1 shows a top-down perspective of an example network without a robot or any viruses. The left-most hallway in this maze (oriented in the up-down direction in the Figure) has a length of four cells. During the challenge, the network will only be presented from the first-person perspective of the robot, and no omniscient top-down view of the network will be available. However, during development, there will be an option available to view the network in a top-down perspective as shown. This will allow teams to debug their code and understand how their robots respond to the mazes.

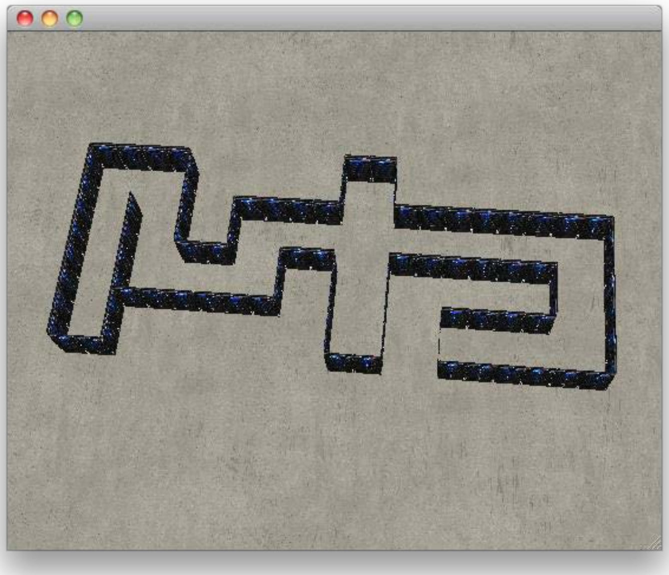


Figure 1: Top-down view of an empty network

As discussed in the following sections, the score awarded for each network is composed of points earned for defeating viruses, collecting packets in the correct order, and cracking the network's secret passphrase.

The Robot

The virtual robot is visualized as a 3D model of a droid, depicted in Figure 2. The robot possesses several sensors:

- three laser ranging sensor capable of detecting the distance to the nearest wall in a particular direction (one sensor each for the front, left, and right directions),
- a pulsed radar sensor capable of detecting the location of any viruses in the robot's immediate vicinity (even if they are behind walls), and
- a pulse radar sensor capable of sensing the locations and IDs of all packets in the network.

Additionally, the robot is capable of stepping forward, stepping backward, turning left, turning right, and jumping (to collect packets). In programming the robot's movement, care should be taken to use the sensors in a manner that keeps the robot from ever running into a wall. The document entitled *Robot Controller Guide* discusses in detail the robot's motion and sensor capabilities as well as how to control the robot programmatically.

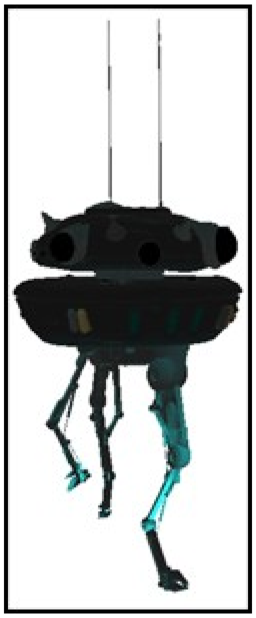


Figure 2: Visual representation of the robot

The Viruses

Each network has been infected with at least one virus. The primary mission of the robot is to locate all viruses and eliminate them. There are two kinds of viruses that may be encountered within this challenge: Benign Bugs and Vile Viruses, shown in Figure 3.

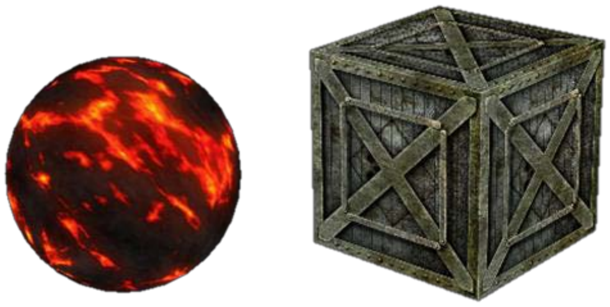


Figure 3: Benign Bug (left), and a lockbox protecting a Vile Virus (right)

Benign Bugs

Benign Bugs are less-invasive viruses that may be found throughout the network. These are the most prevalent of the viruses and are easy to eliminate. To remove a Benign Bug, the robot must simply come into contact with the virus. The collision is enough to eliminate the virus from the network. All viruses are located in the center of a cell within the network. Each defeated Benign Bug is worth **1,000 points**.

Vile Viruses

Vile Viruses are more sophisticated and require an additional step to defeat. Vile Viruses are secured within a lockbox and must be disarmed with a secret code. Upon collision with a Vile Virus, teams are presented with a cryptography puzzle that must be solved to disarm the virus. The cryptography puzzles are discussed in more detail in the section titled Solving the Cryptography Puzzles. When the code is cracked, the lockbox disappears, and the virus is exposed and defeated. Each defeated Vile Virus is worth **5,000 points**.

The Packets

In addition to the viruses, each network includes a set of packets scattered about the maze. The packets are ordered by number, and the robot must collect the packets in the correct order to collect bonus points. At any given time, all but one of the packets will be in a "disabled" or uncollectible state; only one packet will actually be enabled for collection. Collecting an enabled packet removes it from the network and automatically enables the next pack in numerical order. Thus, to collect all of the packets, the robot must first find and collect packet 1 (which is the only enabled packet at the start of the network). Collecting packet 1 removes it from the network and enables packet 2. When the robot finds and collects packet 2, packet 3 becomes enabled. When the robot finds and collects packet 3, packet 4 becomes enabled, and so on. Once the last packet in a network has been collected, 10,000 bonus points are awarded. Figure 4 shows examples of an enabled packet (which is green) and a disabled packet (which is translucent yellow).

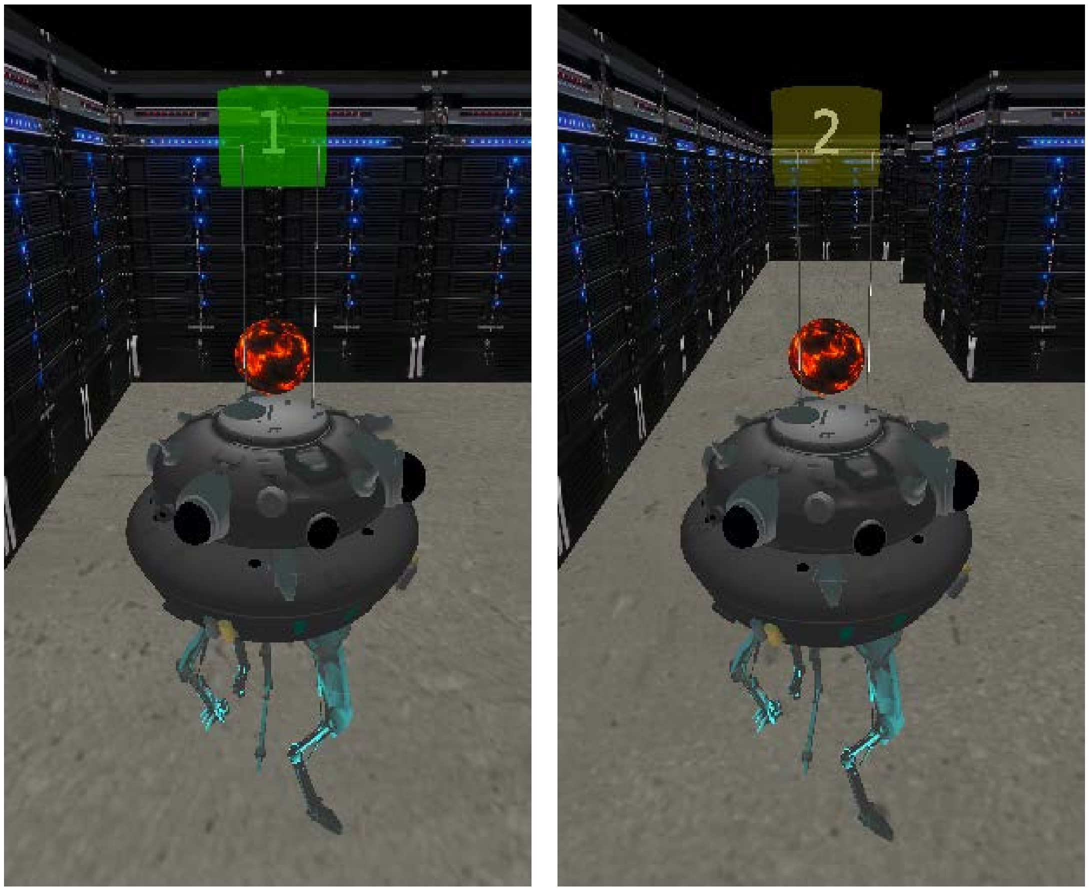


Figure 4: An enabled packet (packet 1, left) is green, while a disabled packet (packet 2, right) is translucent yellow.

The packets are always located above a Benign Bug, although not all Benign Bugs have packets above them. To "collect" an enabled packet, the robot simply needs to move through it. Unfortunately, the packets hover too high above the Benign Bugs for the robot to roll through with its normal motion. Thus, to collect an enabled packet, the robot will need to position itself directly underneath the packet and then *jump* to collect the packet. For more on how to make a robot jump, see the document entitled *Robot Controller Guide.*

The Network Passphrases

As teams watch their robots autonomously navigate through the networks, they may notice text *clues* floating around in various places. Figure 5 shows an example of a clue reading 1210, which means "the base-20 number C". For extra points, teams can write down these clues and use them to work out the secret passphrase for each network. Each network has exactly one passphrase that can be guessed from the main menu at any time during the challenge. Each correctly solved passphrase is worth **10,000 points**. For more on deciphering the clues and solving the network passphrases, see the section titled *Cracking the Network Passphrases* later in this document.

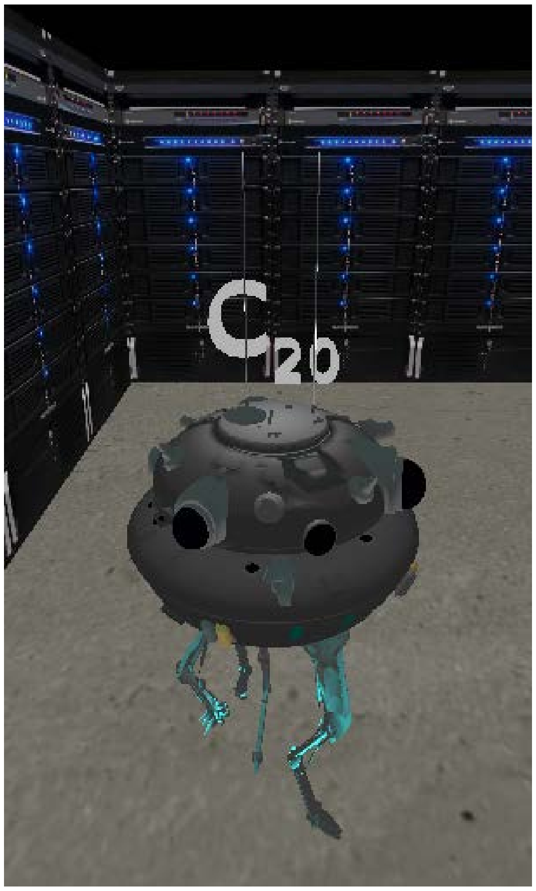


Figure 5: A numerical clue to the network's secret passphrase.

The Challenge: Putting the Pieces Together

The challenge will consist of nine separate networks presented as different levels, each of which can only be attempted if sufficient points have been earned from previous levels. A network is complete when the robot has found and eliminated all viruses in the network. A network may be attempted as many times as desired, and the robot controller may be tweaked if necessary between each attempt. Only the highest score for each network is recorded, so networks can be reattempted without risk of losing points already won during earlier attempts. Networks can be aborted at any time, and the score earned just before the network is aborted is recorded. Figure 6 shows a screenshot of the main challenge menu. All the unlocked networks are shown in light grey with the current high score displayed over the maximum possible score for that network. The locked networks show the score required to unlock the network. The "Guess Phrase" button allows team to try to crack a network's secret passphrase; phrases that have already been cracked have buttons replaced with checkmarks. At the very bottom of the screen, the total score is displayed.

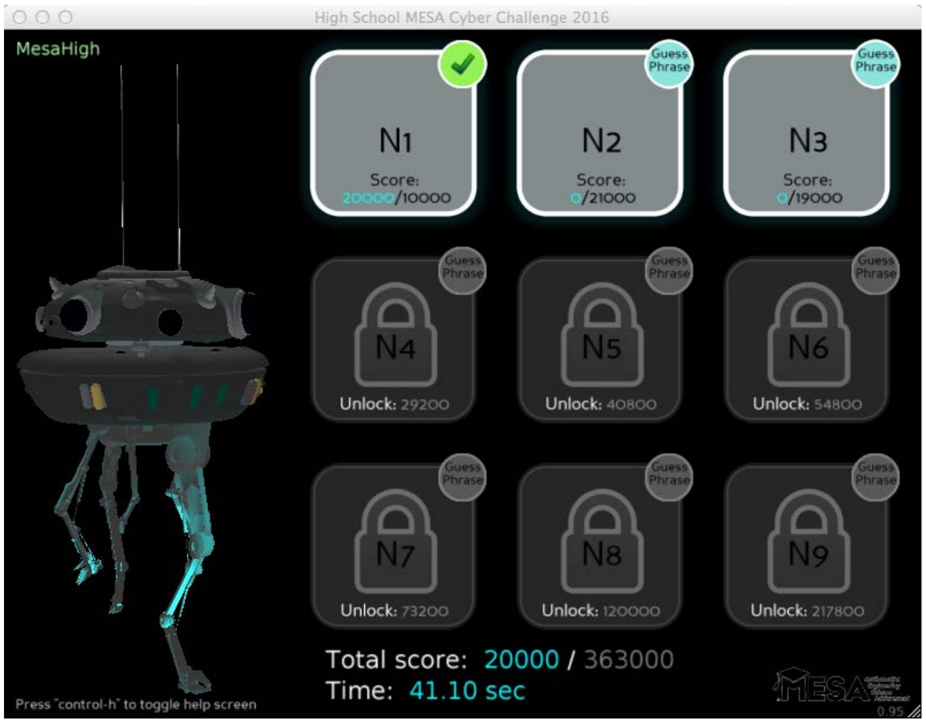


Figure 6: Challenge Menu

The following are the four primary components of the challenge:

1. **Develop and apply autonomous robot controllers.**
2. **Solve the cryptography puzzles to defeat the Vile Viruses.**
3. **Collect packets in the correct order.**
4. **Use the clues to crack each network's passphrase.**

Robot Controller Development

One of the primary challenge components for which teams are responsible is the development of a series of robot controllers. Each network will require its own robot controller because there may be a need to use a different maze-navigating algorithm for each network maze. To encourage code reuse, teams will be able to submit a code base that includes core robot controller functions in a Python module that can be referenced by each particular robot controller. Such a code base, if designed well, can facilitate rapid construction of network-specific robot controllers on competition day. The document entitled *Robot Controller Guide* discusses in detail how to develop a robot controller.

It is important to realize that the networks the robot navigates through on challenge day will be brand new networks not seen before. Thus, the robot controllers ought to be capable of navigating through network mazes without knowing the exact maze structure beforehand. Controllers that can navigate this way will perform well on challenge day; controllers that can navigate *quickly* (without a lot of unnecessary movement in unprofitable directions) this way will perform even better. Teams are encouraged to consider general maze-navigating and on-the-fly map-building techniques when designing their robot controllers.

Solving the Cryptography Puzzles

The second major component of the challenge is solving the cryptography puzzles to defeat the Vile Viruses. Upon colliding with the Vile Virus, teams are presented with a cryptography puzzle. The puzzle text will either be a secret code (ciphertext) that needs to be *decrypted* into plain English text (plaintext), or the puzzle text will be in plaintext and will need to be *encrypted* into ciphertext. The prompt will indicate the goal (decryption or encryption) for the specific cryptography puzzle.

The puzzle will include information that will assist in finding its solution, such as the cryptographic key/cipher to use or a hint about the key. The puzzle will contain a text box into which the user must enter a guess for the puzzle's solution. All guesses are case-insensitive, so it does not matter whether the text is in uppercase or lowercase letters. Ciphertext should be submitted with no spaces (for example, EQORWVGTUEKPEG). Plain text should be submitted *with* spaces in the appropriate places (for example, OCEAN CITY). Any punctuation should be omitted in both the ciphertext and plaintext submissions.

If the puzzle is solved, points are awarded for defeating the virus, and the robot can continue with the remainder of the network. The defeat will be remembered for future attempts of the network if the network is reattempted with changes to the robot controller. There is no limit to the number of guesses that may be made. However, the more time it takes to solve the puzzle, the less time there may be to find more viruses.

In the case in which a puzzle cannot be solved, there are two options. The first option is to skip the virus. If a Vile Virus is skipped, the virus fades and becomes dormant for the remainder of the attempt, and the robot can continue to traverse the rest of the network. Skipping a virus forfeits the points that could have been claimed by defeating the virus, reducing the maximum achievable score for the network. However, if the level is reattempted, the Vile Virus will no longer be dormant, and the cryptography puzzle will be presented again upon collision. The second option if a puzzle cannot be solved is to exit the network and to attempt a different network. The points achieved in a network prior to exiting are recorded.

For background on cryptography and on the various types of ciphers and encryption/decryption algorithms that may be encountered in this challenge, see the document entitled *Introduction to Cryptography.*

The ciphers used specifically in this challenge are the following:

- Caesar Cipher
- Vigenère Cipher
- Transposition Cipher
- Polybius Square Cipher
- Public/Private Key Cipher

The resulting plaintext will always consist of real, comprehendible words. The *Introduction to Cryptography* document will be provided on challenge day. Additionally, all encryption/decryption problems will need to be solved by hand on challenge day (i.e., no calculators or computers). Pencils and blank paper will be provided. ***HINT***: if everyone on the team is familiar with all the ciphers, the entire team can try different approaches in parallel, minimizing the time spent breaking the code.

Collecting the Packets in the Correct Order

While defeating the viruses is the robot's primary mission, an optional side mission is to collect all the network packets in numerical order. Collecting all the packets in a network will results in 10,000 bonus points being awarded for that network. To collect all the packets, the robot must first find, navigate to, and collect packet 1. Then it must find, navigate to, and collect packet 2. Packets are "collected" if they are enabled (i.e., they are the next packet in the sequence) and the robot moves through them. Unfortunately, the packets float high enough above the robot that the robot can't move through them using its normal motion. To collect an enabled packet, a robot must position itself directly beneath the packet and then jump.

It is important to note that the robot will automatically exit the maze immediately upon defeating the last virus in the network—regardless of whether all the packets have been collected or not. Thus, to make sure the bonus points are collected, robots will need to be sure to collect the last packet in the network before defeating the last virus.

Cracking the Network Passphrases

The text clues that can be found throughout the networks are numbers, but unfortunately, they have been encoded in a non-base-10 number system. (For background on converting numbers between different bases, see the document entitled *Introduction to Number Base Conversion*.)

For example, the clue "12_{16}" means "12" is encoded in base 16 (which happens to be the quantity eighteen). Teams must convert these clues back to base 10 before they can make sense of them. Once expressed in base 10, the numbers will be in the range of 0 to 26 and will stand for a letter or space (0 = space, 1 = A, 13 = M, 26 = Z, and so on). After converting the base-10 numbers to letters, teams can unscramble the letters to obtain the network's secret passphrase. Similar to the cryptography puzzles, the number conversions must be done by hand on challenge day.

As an example, consider a run through a network where a team notices the following clues:

31_8 16_9 14_{12} 13_5 40_5 E_{16}

Converting these non-base-10 numbers to base-10 numbers, the team gets:

$31_8 = \mathbf{25_{10}}$ $16_9 = \mathbf{15_{10}}$ $14_{12} = \mathbf{16_{10}}$ $13_5 = \mathbf{8_{10}}$ $40_5 = \mathbf{20_{10}}$ $E_{16} = \mathbf{14_{10}}$

Then, converting these base-10 numbers to letters (1 = A, 13 = M, 26 = Z, etc.), the team gets:

25_{10} = Y 15_{10} = O 16_{10} = P 8_{10} = H 20_{10} = T 14_{10} = N

Finally, by unscrambling the letters YOPHTN, the team finds that the network's secret passphrase is PYTHON. The team can enter the network's secret passphrase to claim bonus points for the network.

Note that teams do not necessarily have to catch and convert every clue for a network to discover the network's secret passphrase. For instance, in the example above, it is possible that teams discovering only the first five letters YOPHT may still correctly guess that the secret word is the six-lettered word PYTHON. Correctly guessing the secret passphrase from just a subset of clues is perfectly acceptable. Also, note that teams are not required to guess a secret passphrase at all: although no extra points will be awarded in such cases, teams will still be free to move on to the next network.

Challenge Engine

The challenge engine serves as the program that runs a team's robots through the networks and ties all the pieces together. To start the challenge engine, the file "challenge.bat" should be executed. If the engine goes down, re-executing this file can always bring it back up. The students should become familiar with running the challenge engine before competition day. Figure 7 shows a screenshot of the challenge engine executing a robot controller.

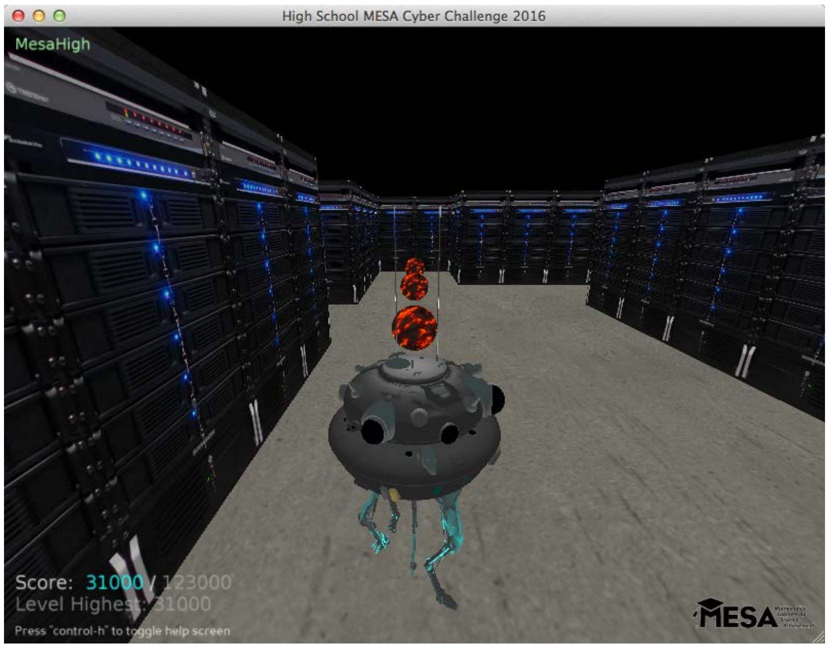
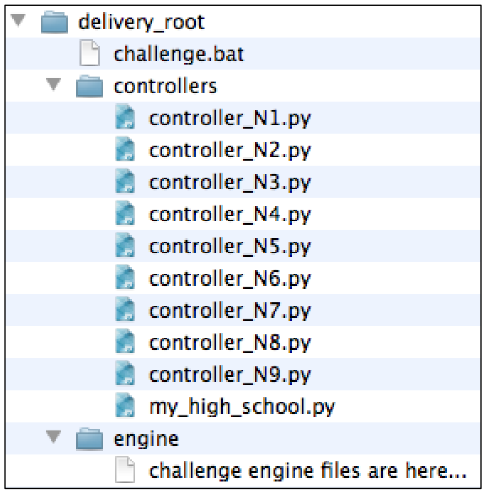

High School MESA Cyber Challenge 2016
MesaHigh
Score: 31000 / 123000
Level Highest: 31000
Press "control-h" to toggle help screen
MESA

Figure 7: Challenge engine execution screenshot

The challenge engine deployment package is delivered with example networks that will be different from (but not unrepresentative of) the networks unveiled on competition day. The example networks are useful for developing and testing robot controller code ahead of the challenge day. Figure 8 shows the directory structure of the files that will be provided in the challenge engine deployment package during competition day.

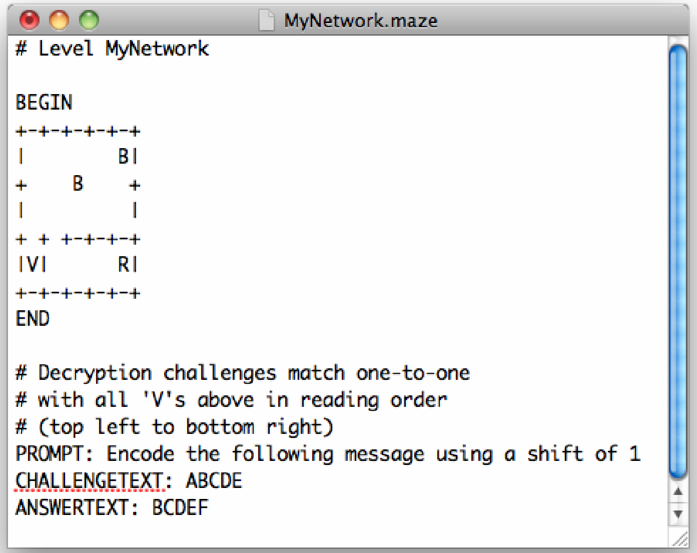


```
delivery_root
    challenge.bat
    controllers
        controller_N1.py
        controller_N2.py
        controller_N3.py
        controller_N4.py
        controller_N5.py
        controller_N6.py
        controller_N7.py
        controller_N8.py
        controller_N9.py
        my_high_school.py
    engine
        challenge engine files are here...
```

Figure 8: Challenge engine deployment package directory structure

Custom Network Mazes (Optional)

If desired, the teams can create their own custom networks. These networks are defined in a text file with a ".maze" file extension that defines what the network looks like, where the robot starts, and where the viruses within the network are located. Teams can also create their own cryptography puzzles and add them to their network maze definition files. Creating custom networks may be a good way to test out the robot controller algorithms on a variety of networks. Figure 9 shows a screenshot of an example maze definition file. Additional documentation on the text file format, along with an example of how to load it into the engine, can be found with the challenge engine deployment package.



```
# Level MyNetwork

BEGIN
+-+-+-+-+-+
|        B|
+    B    +
|         |
+ + +-+-+-+
|V|      R|
+-+-+-+-+-+
END

# Decryption challenges match one-to-one
# with all 'V's above in reading order
# (top left to bottom right)
PROMPT: Encode the following message using a shift of 1
CHALLENGETEXT: ABCDE
ANSWERTEXT: BCDEF
```

Figure 9: Example custom maze

Software Requirements

No external dependencies are required to run the challenge engine other than what is included with the challenge engine deployment package. The engine deploys with an embedded version of Python, which is needed to process the code that is written in the robot controllers.

Although the challenge engine can be executed without having an installation of Python on the machine, it is recommended that an external version of Python be installed for the teams to learn the Python programming language outside of the challenge framework. Because the version of Python used for the challenge is fixed, **it is recommended to install Python 2.5.4**, which can be downloaded from the following webpage:

http://www.python.org/download/releases/2.5.4/

NOTE

If a more recent version of Python is installed, note that some features in the newer version of the Python programming language may not be available within the challenge framework.

The differences between Python 2.5.4 and the latest Python 2.7.x will not likely be observable at the beginner level when learning about the basics of data structures, control structures, and conditional statements, etc. However, if the students start using extended modules or advanced features of the language, the differences may be noticed. This will be especially true of the differences between Python 2.5.4 and any of the 3.x versions of Python.

Thus, to avoid annoying discrepancies between versions of Python teams develop on and the version of Python used on challenge day, **we highly recommend that teams download Python 2.5.4**.

Detailed Specifications and Judging Guidelines

The competition scoring is broken down into four categories:

1. An oral presentation,
2. A written report,
3. The design and implementation of the robot program, and
4. The robot program's performance.

This document has only discussed components (3) and (4) so far. The following guidelines provide details on all four categories.

I. Oral Presentation 25 points

The team must provide an oral presentation describing their approach to solving the problem. The presentation must be between 5 and 10 minutes in length[2] and should be supplemented with a trifold display board. The oral presentation scoring rubric is provided below. The team must address the following topics in its presentation:

1. Overview of the task
   - What problem are you trying to solve?
2. Approach to solving the problem
   - How did you solve the problem?
   - What software algorithms/techniques did you use?
   - What impacted your design decision?
   - How did teamwork play a role in the solution?
3. Challenges encountered
   - What challenges did you face?
   - How did you work around the challenges?
4. Conclusions
   - What did you learn while working on the challenge?

[2] A 5-point deduction will be taken if a presentation is less than 5 minutes or more than 10 minutes long.

II. Technical Paper 25 points

Detailed requirements for the written report are provided on the Scoring Sheet for the Technical Paper. The team's program used during the competition must be that described in the report. Sections should include the following: title page, abstract or summary, introduction, background, discussion, and conclusions. The written report must be received by the competition coordinator no later than 10 days before regional and 10 days before state competition to allow advance time for judging.

The following are the main areas that will be considered in the evaluation of the technical paper.

1. **Discussion**
   - ➢ Summarize the process:
     - ☐ The teamwork – Organizational roles of team to meet the challenge.
     - ☐ Moving through the maze – What do you have to know how to do? How did you learn to do it?
     - ☐ Deciphering the codes – What do you have to know how to do to solve ciphers? How did you learn to do it?
2. **Abstract**
   - ➢ Three-hundred-word limit
3. **Introduction**
   - ➢ Articulate the challenge: What are you trying to do and to solve?
4. **Conclusions and Recommendations**
   - ➢ What did you learn by undertaking the challenge?
     - ☐ What did you do well?
     - ☐ What do you need to improve?
   - ➢ Provide suggestions.
5. **Written Presentation**
   - ➢ Maximum 5 pages + cover page
     - ☐ Cover page with school name, county, names of team members, and team name
     - ☐ Abstract
     - ☐ Double spaced
     - ☐ Font size 12, Times New Roman
     - ☐ Includes bibliographic resources (at least three—for example, MESA website, Thinkquest, etc.)

III. Code Design and Game Performance 25 points

Detailed requirements for the code design and implementation are provided on the Scoring Sheet for Code Design and Game Performance. All of the code used by the team must be written by the team members. The code to be reviewed includes all code written before the competition day. The code written on challenge day is not included in the review. The code should be written as generically as possible, keeping in mind that the aim is to design an autonomous robot. Any changes made to the code on challenge day are not incorporated into the design and implementation score. The submitted Python code will be graded according to the following:

1. **Demonstrate understanding of robot interface/commands:**
   - Make use of robot sensors
   - Make use of robot motion commands
   - Algorithm logic within a loop structure for continuous behavior (i.e., for loop, while loop)
2. **Algorithm design:**
   - Logical elegance of solution
   - Use of data structures (i.e., variables, lists, dictionaries) to drive robot behavior and decisions
3. **Algorithm efficiency:**
   - How well does the robot navigate the network?
   - Does the robot avoid cells that it has already visited?
   - Does the robot minimize the number of sensor calls it makes?
4. **Algorithm correctness:**
   - Does the algorithm handle all cases appropriately?
   - Does the robot avoid hitting walls?
   - Does the robot avoid getting caught in infinite loops?
5. **Organization:**
   - Well-used data structures
   - Good functional breakdown
   - Minimal code duplication (code reuse)
   - Minimize global variables
6. **Coding style:**
   - Descriptive variable names
   - Useful comments
   - Consistency
7. **Effort**

Tie-Breakers

Ties will be broken via the use of the highest score on the following rubric indicators:

1. **API Comprehension**
2. **Algorithm Design**
3. **Algorithm Efficiency**

IV. Performance Demonstration (Maze Score) 25 points

During the maze execution part of the competition, the challenge engine will keep track of the score, which is the sum of the points accrued from the viruses and the clues. The amount of time it has taken to accrue the current score is also tracked and is used for tiebreakers. This means that if two teams have the exact same score, then the team that got that score faster is scored more favorably. The total score is then scaled to a number between 0 and 25, which is used as the points for the performance portion of the competition.

ALL DECISIONS MADE BY JUDGES ARE FINAL

Cyber Robot Challenge **Scoring Sheet**

MARYLAND MESA Mathematics Engineering Science Achievement JOHNS HOPKINS UNIVERSITY APPLIED PHYSICS LABORATORY

School: ________________________ Judge: ________________________ Total __________ / 25 points

Performance Area	Level of Mastery (Select One)					Total
Oral Presentation	**None**	**Developing**	**Approaching**	**Some**	**Mastery**	
Subject Matter/Content Discussion						
Student(s) significantly increase the judges' understanding of the cyber robot challenge task and their team's approach to solving the presented problem(s)—including but not limited to the following:	***Section Total 5 Points***					________/5
➢ Valid explanation with supporting details/examples.	(0.2)	(0.4)	(0.6)	(0.8)	(1)	________/1
➢ Evidence of purposed attention to the quality of discussion.	(0.2)	(0.4)	(0.6)	(0.8)	(1)	________/1
➢ Sharing of any challenges and how they were overcome.	(0.2)	(0.4)	(0.6)	(0.8)	(1)	________/1
➢ The team introduced all of the team members and detailed the contributions of each.	(0.2)	(0.4)	(0.6)	(0.8)	(1)	________/1
➢ Compelling explanation that provides judge(s) vivid clarity on the team's thought process towards engineering & design.	(0.2)	(0.4)	(0.6)	(0.8)	(1)	________/1
Demonstration of Skill						
Student(s) present using highly effective speaking skills to clearly convey their message—including but not limited to the following:	***Section Total 10 Points***					________/10
➢ Professionalism: Clear speech (avoids "um," "like," "you know"), eye contact, professional posture, etc.	(0.5)	(1)	(1.5)	(2)	(2.5)	_______/2.5
➢ Speech at a pace that does not detract from the message.	(0.5)	(1)	(1.5)	(2)	(2.5)	_______/2.5
➢ The team delivered a brief, engaging introduction, a uniquely interesting presentation, and a highly compelling conclusion.	(0.5)	(1)	(1.5)	(2)	(2.5)	_______/2.5
➢ Team demonstrated that they have command knowledge of the information they presented, showing an ability to readily respond to questions from the judges.	(0.5)	(1)	(1.5)	(2)	(2.5)	_______/2.5
Delivery/Awareness						
Student(s) present creatively using highly engaging elements to maintain the judge's attention and transfer their excitement about the subject matter while conveying their message—including, but not limited to the following:	***Section Total 10 Points***					________/10
➢ Pleasant demeanor.	(0.5)	(1)	(1.5)	(2)	(2.5)	_______/2.5
➢ Appropriate use of visual aids.	(0.5)	(1)	(1.5)	(2)	(2.5)	_______/2.5
➢ Enthusiasm for the subject evident in voice inflection/delivery.	(0.5)	(1)	(1.5)	(2)	(2.5)	_______/2.5
➢ Maintains judge's attention throughout the presentation.	(0.5)	(1)	(1.5)	(2)	(2.5)	_______/2.5
Overall Score for Oral Presentation						
☐ ***If team presents under 5 or over 10 minutes, assess a minus-5-point penalty:***						-________
TOTAL ORAL PRESENTATION SCORE (x/25 points):________/25						

Cyber Robot ChallengeScoring Sheet

School ———————————————— Judge:______________________________ Total_____________./ 25 points

Judge's Feedback, Oral Presentation

Cyber Robot Challenge **Scoring Sheet**

School: ______________________ **Judge:** ______________________ **Total**____________**/ 25 points**

Performance Area	Level of Mastery (Select One)					Total
Technical Paper	None	Developing	Approaching	Some	Mastery	
Discussion (Background Information and Design and Implementation)						
Student(s) provide a thorough discussion of physics, math, and/or engineering concepts, including advanced concepts (if used). Student(s) provide a very clear explanation of program design and implementation—including but not limited to the following:	***Section Total 10 Points***					________/10
➢ Valid explanation with supporting details/examples.	(0.5)	(1)	(1.5)	(2)	(2.5)	_______/2.5
➢ Roles and contributions discussed for each team member.	(0.5)	(1)	(1.5)	(2)	(2.5)	_______/2.5
➢ Sharing of any challenges and how they were overcome (Ex: How did you sole the ciphers?).	(0.5)	(1)	(1.5)	(2)	(2.5)	_______/2.5
➢ Compelling explanation that provides judge(s) vivid clarity on the team's thought process toward engineering and design.	(0.5)	(1)	(1.5)	(2)	(2.5)	_______/2.5
Abstract and Introduction						
Student(s) clearly restate the purpose, key features, and conclusions of the report readily engaging the reader while conveying relevant information—including, but not limited to the following:	***Section Total 5 Points***					________/5
➢ The abstract conforms to the 300-word maximum length.	(0.2)	(0.4)	(0.6)	(0.8)	(1)	_______/1
➢ Information is conveyed in a professional and engaging manner.	(0.2)	(0.4)	(0.6)	(0.8)	(1)	_______/1
➢ Work demonstrates command knowledge of the information presented.	(0.2)	(0.4)	(0.6)	(0.8)	(1)	_______/1
➢ Text provides detailed description of parameters, methods, limiting factors, and technical terms used.	(0.2)	(0.4)	(0.6)	(0.8)	(1)	_______/1
➢ Paper provides explanation for the selected design and problem-solving approach.	(0.2)	(0.4)	(0.6)	(0.8)	(1)	_______/1
Overall Written Presentation						
Student(s) professionally organize the paper providing all key sections—including but not limited to the following:	***Section Total 10 Points***					________/10
➢ Title page, Abstract, Introduction, Discussion, and Conclusions and Recommendations.	(0.5)	(1)	(1.5)	(2)	(2.5)	_______/2.5
➢ Font size 12-pt Times New Roman, Length: 3–5 pages, 1" margins, + cover page	(0.5)	(1)	(1.5)	(2)	(2.5)	_______/2.5
➢ All supporting sections are included: reference, acknowledgements, etc. (at least three)	(0.5)	(1)	(1.5)	(2)	(2.5)	_______/2.5
➢ Proper grammar, spelling, and sentence structure used throughout the paper.	(0.5)	(1)	(1.5)	(2)	(2.5)	_______/2.5
Overall Score for Technical Paper						
☐ ***If submitted late, assess a minus-5-point penalty:***						-_______
TOTAL TECHNICAL PAPER SCORE (x/25 points):________/25						

Cyber Robot Challenge **Scoring Sheet**

School: ______________________ **Judge:______________________** **Total__________/ 25 points**

Judge's Feedback, Technical Paper

Cyber Robot Challenge **Scoring Sheet**

School: ______________________ Judge: ______________________ Total____________/ 25 points

Performance Area	Level of Mastery (Select One)					Total
Code Design and Game Performance	None	Developing	Approaching	Some	Mastery	
API Comprehension: Use of Robot Commands / Use of Robot Sensor Commands: TIE-BREAKER #1						
Student(s) demonstrate a strong use of robot commands and robot sensor commands—including but not limited to the following:	***Section Total 7.5 Points***					________/7.5
➢ Motion commands are clearly understood and are strategically called within an overarching loop for continuous robot behavior. (Robot also avoids hitting walls.)	(0.75)	(1.50)	(2.25)	(3.00)	(3.75)	______ /3.75
➢ Sensors are being used to determine where the robot is to go. The results of the sensor calls feed into the motion commands.	(0.75)	(1.50)	(2.25)	(3.00)	(3.75)	______ /3.75
Algorithm Design: Accuracy, Elegance, and Use of Data Structures TIE-BREAKER #2						
Student(s) display an accuracy and elegance in design with an effective use of data structures—including but not limited to the following:	***Section Total 3 Points***					________/3
➢ A solution that is logically elegant, never resulting in wall collisions, and enables robot to traverse entire network with a single generic algorithm.	(0.2)	(0.4)	(0.6)	(0.8)	(1)	_______/1
➢ Algorithm is clean, systematic, and easily extensible. Logic is not unnecessarily complicated. (Commands are not hard-coded).	(0.2)	(0.4)	(0.6)	(0.8)	(1)	_______/1
➢ Data structures are efficiently used to map entire network (visited cells and/or sensor results) and drive robot decisions.	(0.2)	(0.4)	(0.6)	(0.8)	(1)	_______/1
Algorithm Efficiency: Network Traversal / Sensor Calls / Software TIE-BREAKER #3						
Student(s) work in network traversal optimally utilizes sensor calls and software—including but not limited to:	***Section Total 3 Points***					________/3
➢ Algorithm prevents robot from visiting cells that it has already visited to the greatest extent possible.	(0.2)	(0.4)	(0.6)	(0.8)	(1)	_______/1
➢ The number of sensor calls is minimized by tracking which cells have been sensed and by not repeating sensor calls.	(0.2)	(0.4)	(0.6)	(0.8)	(1)	_______/1
➢ Software is written efficiently and there is little to no room for optimization.	(0.2)	(0.4)	(0.6)	(0.8)	(1)	_______/1
Coding Style and Organization: Functional Breakdown/Function, Variable, Class Naming Convention/ Readability/Comments						
Student(s) produce code with optimization in functional breakdown, variables, and class naming convention with excellent readability; including but not limited to the following:	***Section Total 4 Points***					________/4
➢ No code duplication and logical and efficient functional breakdown.	(0.2)	(0.4)	(0.6)	(0.8)	(1)	_______/1
➢ Names selected are descriptive, clear, and appropriate.	(0.2)	(0.4)	(0.6)	(0.8)	(1)	_______/1
➢ Code is extremely self-descriptive and its intention is very easy to comprehend.	(0.2)	(0.4)	(0.6)	(0.8)	(1)	_______/1
➢ Extensively commented where code is not self-descriptive. Functional inputs and outputs are well commented	(0.2)	(0.4)	(0.6)	(0.8)	(1)	_______/1
Continued…						

School: ____________________________________ Judge:_______________________________ Total____________/ 25 points

Performance Area	Level of Mastery (Select One)					Total
Code Design and Game Performance (Continued)	None	Developing	Approaching	Some	Mastery	
Effort: Python Software/ Maze Traversal						
Student(s) demonstrate noteworthy effort in mastering the Python programming language—including but not limited to the following:	***Section Total 7.5 points***					________/7.5
➢ Excellent effort in using the Python programming language. Clear indication that Python programming references were consulted to understand data structures, control structures, organization, etc.	(0.75)	(1.50)	(2.25)	(3.00)	(3.75)	______ /3.75
➢ Strong understanding of the concepts for autonomous traversal are clearly seen	(0.75)	(1.50)	(2.25)	(3.00)	(3.75)	______ /3.75
Overall Score for Code Design and Game Performance						
☐ ***If submitted late, assess a minus-5-point penalty:***						-_______

TOTAL CODE SCORE (x/25 points):________/25

TOTAL MAZE SCORE = GAME PERFORMANCE (x/25 points):__ _/25

OVERALL SCORE (x/100 points)______/100

Scoring Sheet

School: ______________________________ Judge: ______________________________ Total__________/ 25 points

Judge's Feedback, Code Design and Game Performance