National Aeronautics and Space Administration

Drones to the Rescue!

NASA Office of STEM Engagement Next Gen STEM

EDUCATOR NOTES

Drones to the Rescue

Learning Objectives

Students will:

- Use coding to replicate a firefighting simulation
- Evaluate other student projects to determine how well they met the criteria and constraints of the problem

Investigation Overview

- This activity introduces students to coding, or you can use it as a culminating activity after students have learned about the various coding blocks. Additionally, this activity supplements NASA's "**Package Delivery Drone Simulation**" activity. The original activity introduces students to basic coding to create an interactive simulation of a drone navigating around a geofenced area to deliver a package. The "Additional Resources" section at the end of this activity also contains other NASA products to practice block-based programming.
- For this activity, students use ***Scratch, Snap!***, or another programming language to create an interactive simulation of a drone navigating within a geofenced area (simulated wildfire) to help crews on the ground fight a wildfire. The simulation engages students in computational thinking, problem solving, and real-world application of mathematics.

Package Delivery Drone Simulation

NASA CONTEXT

Each year, thousands of wildfires release large amounts of planet-warming carbon dioxide into the atmosphere and burn roughly 1.5 million acres of forests and grasslands throughout the country, according to the U.S. Department of Agriculture Forest Service. Wildfires impact all living things — humans, animals, plants, and trees and this activity gives students an opportunity to

Photo - John McColgan BLM Alaska Fire Service

MIDDLE SCHOOL

Grades 6-8

Suggested Pacing

60 to 120 minutes

Materials

- Computer or tablet with internet access (or download all content to run locally).
- Download file containing: Drone Sprites, Stages, and Geofenced Area Sprites.
- Programming language of your choice (Note: *Snap!* and *Scratch* can be downloaded to run locally) This activity gives instructions using *Scratch*
  - Free *Scratch* account at: **http://scratch.mit.edu**
  - Free *Snap!* Account at: **https://snap.berkeley.edu**

National STEM Standards

- **MS-ETS1-2 Engineering Design**
  Evaluate competing design solutions using a systematic process to determine how well they meet the criteria and constraints of the problem.
- **MS-ETS1-3 Engineering Design**
  Analyze data from tests to determine similarities and differences among several design solutions to identify the best characteristics of each that can be combined into a new solution to better meet the criteria for success.
- **MS-ETS1-4 Engineering Design**
  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.

see how they could help. Suppressing these blazes is a complex and costly operation — with suppression costs averaging $2.9 billion over a five-year period. Containing and responding to these fires also requires collaboration among firefighters, ground crews, and various local, state, and federal agencies. It also involves the coordination of dozens of aircraft operated by multiple government agencies. It is also important to note that firefighting does not always involve suppressing fire, but sometimes agencies use fire in a controlled manner to prevent unintentional wildfires that could be started by lightning or human activity by burning underbrush and dried grass that could be easily set ablaze.

NASA's **ACERO** (Advanced Capabilities for Emergency Response Operations) project — led by the agency's Ames Research Center in Silicon Valley, California — is developing innovative aviation and communication technologies to support a new era of wildland fire management. Using ACERO technologies, uncrewed vehicles, or drones, could be safely integrated into firefighting operations – helping suppress wildland fires and adding valuable operational support. At night, or during low visibility, drone operators could become the "second shift" of first responders to remotely fight fires well beyond what is currently possible. Drones could also provide continuous monitoring and communication services and operate like cell towers in the sky. Thermal sensors onboard the drone could map the location of a fire, its size, and how fast it is growing, and this information can be transmitted in real time to firefighters. ACERO technologies could also support first responders during other natural disasters like hurricanes, floods, and tornadoes.

In this activity, students will use drones to build a game within a coding platform to fight a simulated fire. Students will use the "stage" file provided with this activity to build their game upon. The image shows a landscape containing a "pre-fire," an "active fire," and a "post-fire" area. Students will use the provided drone "sprites" to navigate within a geofenced area representing the active fire and code messages to the ground crew of where the dangers lie. As a note, using drones to fight wildland fire is a huge advantage as they are able to fly above the fire and become "eyes in the sky" for ground crews. Within the geofenced area, students will also place obstacles the drone must navigate around, such as a cell tower. A search and rescue helicopter and a tanker plane are other provided sprite files. Manned aircraft always take precedence over unmanned vehicles, so it is important the drones do not collide with these vehicles, and it is imperative the helicopter and tanker plane remain outside the geofenced area.

- If students complete the task with time remaining, you can assign them additional higher-level requirements to keep them challenged (see the "Consider the Consequences" section below). Additionally, they can develop their own requirements and change their code to meet the new requirements.

Introduce the Investigation

- Consider introducing students to NASA's work helping to combat wildfires with the following resources:
  - **Video – Operation Earth: Wildfires** (28:23)
    https://climate.nasa.gov/climate_resources/201/video-operation-earth-wildfires/
  - **Video – Wildfires 101: How NASA Studies Fires in a Changing World** (5:05)
    https://www.youtube.com/watch?v=I9YO0LBorMg
  - **Video – Are Wildfires getting Worse? We Asked a NASA Scientist** (1:42)
    https://www.youtube.com/watch?v=1m37B2MJKu8
  - **Podcast – NASA's Curious Universe: Wildfires From Space**
    https://www.nasa.gov/podcasts/curious-universe/wildfires-from-space/
- Block-based coding simplifies the process of writing computer programs, making programming more accessible to students. In block-based coding, each block contains a program component, and students combine blocks using a drag and drop editor. By selecting different combinations of coding requirements, you can modify the activity based on the skill level of the students and the lesson's focus. This activity can be done individually by students, or you can pair them, based on the educator's discretion. For educators or students who are true beginners to block-based programming, a step-by-step guide using *Scratch* has been provided at the end of this activity.
- Programs in *Scratch* have three main parts: stages, sprites, and scripts. A stage is the background picture for the program, while sprites are other objects placed over the stage. The script(s) control the motion and interaction of the sprites.
- Students should use the ball sprites included in *Scratch* as the start/end point, the delivery location (where to send communication to the firefighters on the ground), and the waypoints (each stop a drone should make during the game). You can adjust the size of these sprites in the programming environment.
- Waypoints control the drone's flight by serving as an endpoint for a portion of the flight path. More than one waypoint may

be required to successfully maneuver within the geofenced area and around an obstacle.

- Students will be assigned a stage for their program. They must determine which area(s) within a stage to make available to drones; then, they must create a geofence, or virtual barrier, around that chosen area(s).
- Students choose one of the drone sprites to move around in their program.
- Students must determine a location within the geofence to place an obstacle; then they must choose one of the obstacles sprites and place it in that location. Any of the shapes provided by *Scratch* can suffice as an obstacle.
- Students can use any combination of the geofence sprites to create the barrier around the area of the stage they choose to operate drones within. Initially, the barrier should be blue. If the drone flies out of the area, the barrier should turn red.

Criteria	Constraints
Use stage provided	The drone cannot exit a geofenced area
Create begin/end using ball sprites	
Create waypoints for the drone	
Create a geofence around the fire	
At least one obstacle must be placed within the geofenced area	

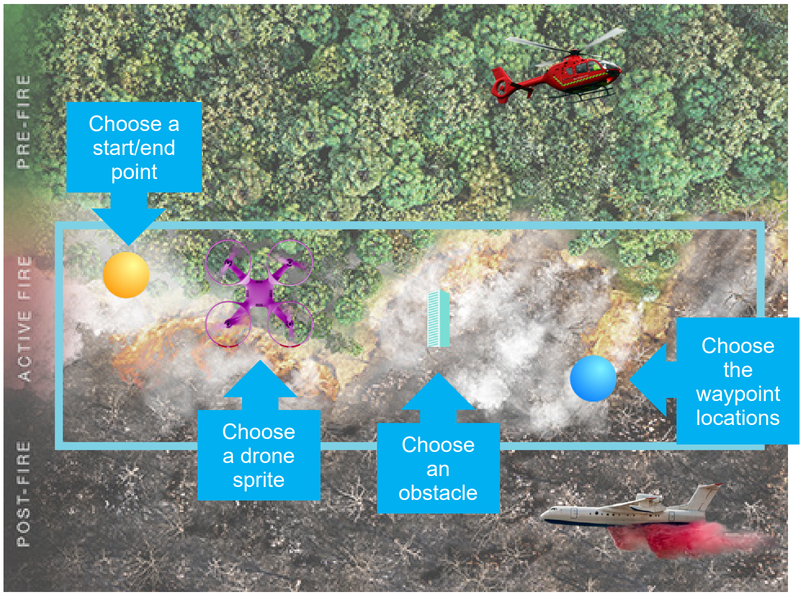
Facilitate the Investigation

Meet the Problem

- For this activity, upload the stage provided for your program. On the stage, choose what area(s) of the fire are available for the drone to survey and create barriers, or geofences, around them. Pick a start/end point for the drone and a location where it will fly to communicate with the firefighters on the ground. The flight path must take the drone completely around at least one obstacle.
- Add an obstacle the drone must avoid by placing an obstacle sprite within the geofenced area (provided by *Scratch*). If the drone collides with the obstacle, the program should end.
- To create the drone's flight path, add as many waypoints as needed. The drone will fly from the start point to the first waypoint, then to the second, and so on. One of the waypoints should be the communication location. The drone will end its flight back at the start/end point.
- If the drone flies out of a geofenced area, the geofence should turn from blue to red. Also, the drone should immediately stop flying.

Explore the Knowns and Unknowns

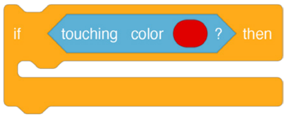
- Upload the stage provided
- Choose a drone sprite
- Choose what area(s) to make available to drones
- Add geofence sprite(s) to create a barrier around the drone-flying areas
- For each geofence sprite, add a costume to enable it to change color
- Choose the obstacle location and add an included shape sprite, such as a building
- Choose the start/end point and add a sprite (the included ball sprite works well)
- Choose the communication location and add a sprite (the included ball sprite works well; use a different color than the start/end point).
- Choose appropriate sizes for the sprites. The drone will not be to scale with the background, so scale it to the appropriate size.
- Add additional waypoints as necessary to guide the drone within the geofenced area and around the obstacle (the included ball sprite works well; use a different color than the other ball sprites).



- Write your script so the drone moves from waypoint to waypoint to deliver the communications to the ground crew and return to the start/end point
- Pause at the communication location and alert the firefighter the drone is dropping off the message
- The flight path takes the drone completely around at least one obstacle
- Add a script to turn geofence sprites red and stop the program if the drone exits the geofenced area

Generate Possible Solutions

- Add additional waypoint(s)
- Add manual controls so the user can control the drone's movement
- Add a button the user can click for directions
- Instead of using the geofenced sprites, draw the geofences on the background stage and use the "touching color" block to detect if the drone is touching the geofenced area



```
if touching color ● ? then
```

Consider Consequences

You can extend this activity in many ways for students who quickly grasp the concept:

- Add a splash screen at the start of the program to give the basic instructions
- Use the zip files of the tanker or the search and rescue helicopter and program it to fly around outside the geofenced area. If the aircraft encounters the geofence sprite, the color should change from blue to red, and the program should immediately stop.
- Create a new sprite for the drone using *Scratch's* paint tool or other software
- Add sound effects (there are sound effects built into *Scratch*, or students can record their own)
- Create a new background (stage) using mapping software. This could include the area around where the student lives.
- The program determines and records whether the drone delivered the communication message to the correct delivery location (i.e., did it touch the delivery location sprite during the drop-off?)
- Add multiple delivery locations

Present Findings

Instruct students to trade their coding simulation with other students and have them assess each other's code by answering the following questions:

- Does the program meet all the assigned requirements?
- Does the drone do what it is supposed to do with no errors?
- Did the student go beyond the basics in creating their program?

Reference

- Package Delivery Drone Simulation Coding Activity: **STEM LEARNING: Package Delivery Drone Simulation Coding Activity Guide (nasa.gov)**

Additional Resources

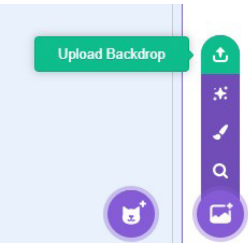
- Attack of the Drones: **https://www.nasa.gov/stem-content/attack-of-the-drones/**
  - NASA's Aeronautics Research Mission Directorate developed this tutorial-style activity to guide students through the creation of a side-scrolling game which highlights some of the aspects of AAM (Advanced Air Mobility).
- Explore Mars with *Scratch*:
  - **https://www.jpl.nasa.gov/edu/teach/activity/explore-mars-with-scratch/**
  - NASA's Jet Propulsion Laboratory in Southern California designed this activity, which takes a student through the steps of creating a simple game where the user controls a rover on Mars.
- Crew Orbital Docking Simulation:
  - **https://www.nasa.gov/stem-ed-resources/crew-orbital-docking-simulation-coding-sim.html**
  - Designed as part of the Commercial Crew Program, this NASA activity challenges students to create a simulation of a commercial spacecraft docking with the International Space Station.
- Operation Earth: Wildfire Activity for a deeper dive into NASA's work with wildfires:
  - **https://mynasadata.larc.nasa.gov/lesson-plans/operation-earth-wildfires**
- **Code.org** for beginner tips/tricks on coding basics.

Setting up the Programming Environment

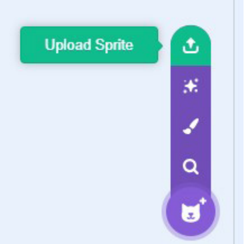
Beginner:

1. **Create a *Scratch* account:** If working in the cloud, students need to create individual or team account(s) at **http://scratch.mit.edu**. This is where they sign in and create a new project. Alternatively, *Scratch* can be downloaded and run locally on a computer. This way, students can work on projects without an active internet connection.

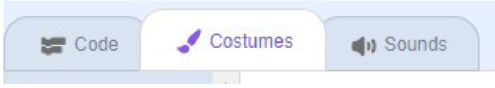
   If *Snap!* is preferred over *Scratch* as the programming environment, it can be accessed at **https://snap.berkeley.edu**. As with *Scratch*, students need to create an account on *Snap!* and it can be downloaded and run locally.
2. **Download and unzip files:** The zipped file contains the files needed for this activity. The contents of this zipped file are shown in the "Block-Based Coding" section of this guide.
3. **Select and add a stage:** Upload the stage found in the zipped file. The "Upload Backdrop" button is in the lower-right section of the *Scratch* programming environment.



4. **Add sprites:** Delete any sprites that are in the program by clicking on the garbage can button next to the sprite. Then, click the "Upload Sprite" button to add a drone sprite. Use the "Upload Sprite" button to also upload any blue geofence sprites and as many waypoints as you need (use the ball sprites in *Scratch* for the waypoints).

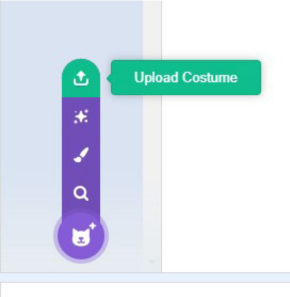


5. **Edit sprites' sizes and/or direction:** Drag the sprites to where you want them on the stage. For each sprite, you can adjust the size to ensure all sprites are relatively proportional in size to one another. You can also click on the "Direction" box and rotate sprites. This may be necessary to align the geofence sprite with the area(s) you want to be off limits.
6. **Add costume(s):** Geofences will initially be blue. If the drone flies into a geofenced area, the geofence sprite will need to turn red. To do this, you need to upload a costume.



   Once you select the sprite in the bottom right portion of the screen, you need to click the "Costume" tab. Then, click the "Upload Costume" button and add the corresponding red version of the geofence sprite.

   When writing scripts, you can change the costume at any time. So, when the script detects the drone has run into the geofenced area, the geofence sprite is changed to red.

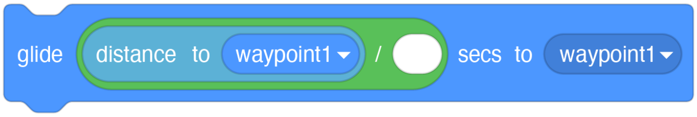


Coding Tips

1. **Positioning and moving sprites:** Locations on the stage are indicated using a set of coordinates, like graphing points on a piece of graph paper. The origin has the coordinates (0, 0) and is in the center of the stage.

   You can use the *Scratch* commands "go to..." and "glide to..." to move your drone sprite to the location of another sprite. This is where the waypoints, which are sprites, are useful — they allow you to move your sprite from point to point to determine the flightpath.

   To control the speed of the sprite, you can use:



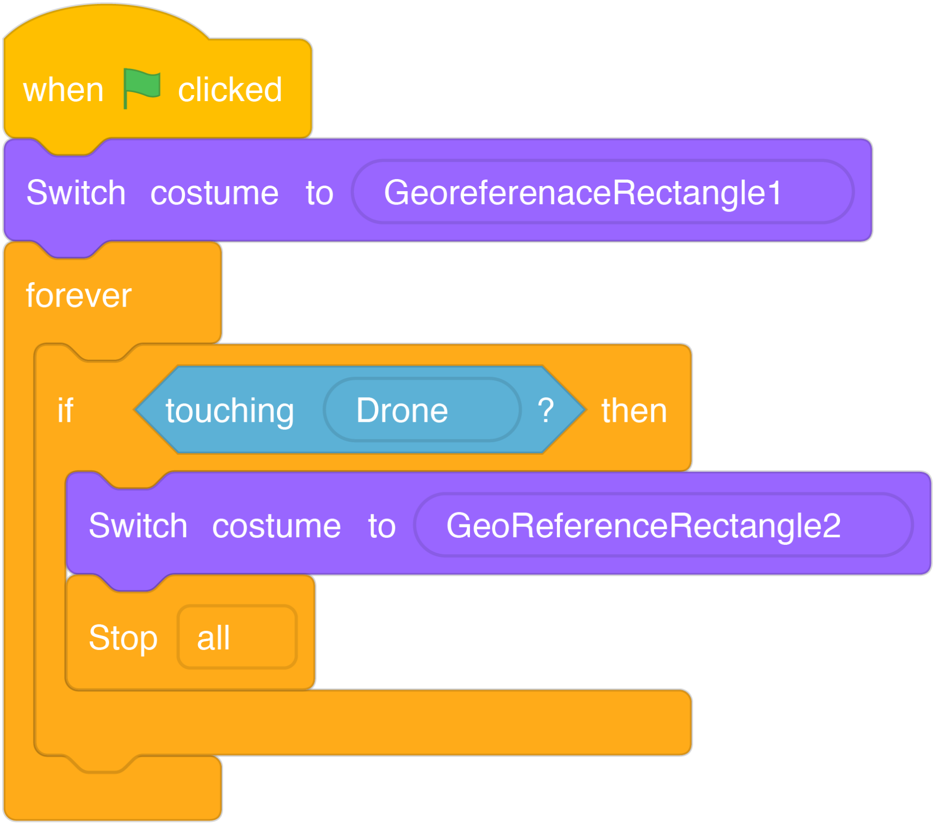
```
glide (distance to waypoint1 / ( )) secs to waypoint1
```

   Scratch calculates the distance the drone must move to a point (waypoint1 in this example). Dividing this distance by the desired speed gives the number of seconds needed and helps control the sprite's speed.

2. **Using costumes:** Costumes in *Scratch* are alternate forms of sprites. For example, if you have a blue box representing the border around a geofenced area, you can replace it with a red box to show that the drone has hit the geofence.
   To do this, you need to first upload whichever shape geofence sprite you need. Then, select that sprite and click the "Costumes" tab in the upper left portion of the programming environment. That is where you upload the alternate version. It is important that both costumes are the same shape.

3. **Detecting collisions between sprites:** An important aspect of this activity is detecting when the drone sprite contacts the geofence sprite. *Scratch* has commands to help with this. To detect a collision, you need to add a script for the geofence sprite.

This script should begin when your program does. First, it changes the geofence costume to the blue outline. Then it continuously checks to see if the drone sprite is touching the geofence sprite. If so, it changes the costume so the geofence is red. It also stops the program at that point, so the drone cannot leave the geofenced area.



```
when ⚑ clicked
Switch costume to GeoreferenaceRectangle1
forever
    if touching Drone ? then
        Switch costume to GeoReferenceRectangle2
        Stop all
```

For more, join our community of educators, NASA CONNECTS!

https://stemgateway.nasa.gov/connects/s

National Aeronautics and Space Administration

NASA Headquarters
300 E Street SW
Washington, DC 20546
www.nasa.gov/centers/hq

www.nasa.gov

NP-2024-07-3268-HQ