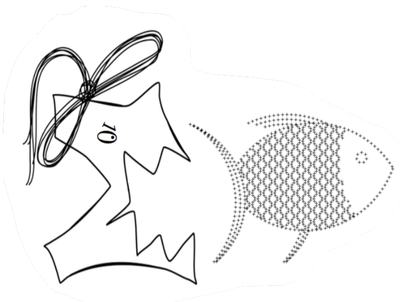
Adventures in 3D: Creating and Delivering Tools for 3D Remote Camps for Middle Schoolers



Larry Yao, Jiwoo Lee, Sharon Wu, Dennis Wang, Christopher William Schankula, Christopher K. Anand[†], and McMaster Start Coding Team

{yaol13, leej229, wuy324, wangc66, schankuc, anandc}@mcmaster.ca and mcmasteroutreach@gmail.com

[†]Department of Computing and Software, McMaster University http://outreach.mcmaster.ca

August 19, 2020

Introduction

Over the last decade, McMaster Start Coding has taught over 15,000 students Computer Science and has created new frameworks and tools to help K-12 students learn. In response to the COVID-19 pandemic, we developed and provided several virtual camps delivered for free to students through the summer using the Zoom platform and our online learning platform MacOutreach.Rocks.

3D Object API

An Elm-based application programming interface (API) was created to mimic the 2D `GraphicSVG` API our campers learned previously, allowing them to easily learn the API. Thus, we minimized the API learning curve and students instead focused on learning how to think three dimensionally. Some example functions and their type signatures:

- `cube : Float -> Material coordinates -> Object coordinates`
  - An example of a "Mold" for creating 3D objects.
  - Only the size (Float) is given initially; it is then piped into another function to apply the material
- `matte : Color -> Mold coordinates a -> Object coordinates`
  - One of our functions for applying a material to a `Mold`. We have many different molds, including the cube function above. This one applies a solid colour, non-reflective material.
  - The colour for this function comes from the `elm-color` module, rather than `GraphicSVG`.
- `move3D : Dimension -> Object coordinates -> Object coordinates`
  - One of our functions for transforming objects. Others include rotation and scaling.
  - `Dimension` is a type alias for `(Float, Float, Float)`. For simplicity, all units are in centimetres.

3D Bee Simulator Game

For the first 3D camp, campers were placed in groups to design a 3D bee pollination simulator. Campers were tasked with designing both assets (including bees, flowers and hives) for the game and the layout of the level. The teams' levels were then combined into a large game at the end of the week. Here are some examples of assets the campers made for the game:

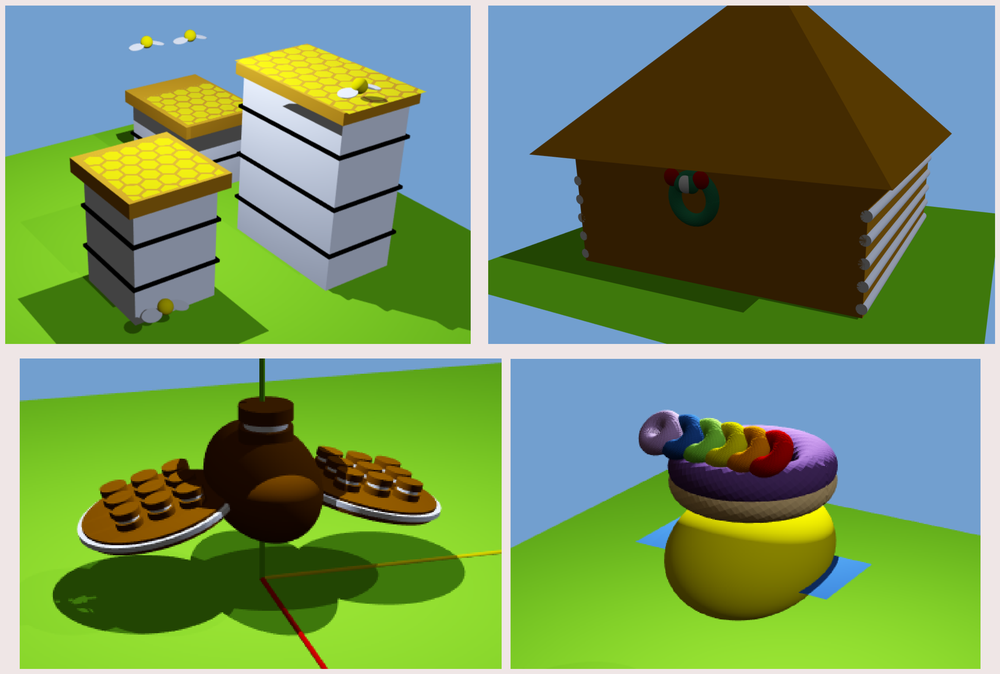


Figure 1: Assets created by middle school students in the camp (shared with permission). Top left: A set of beehives with bees flying around; top right: a gingerbread house; bottom left: a bee made out of cookies; bottom right: a bee made out of doughnuts. One of the teams in the camp decided to have a candy theme for their area in the game.

Open Source Modules Used

Several, fully open source modules were used in the creation of both activities:

- `elm-3d-scene` provides an interface to WebGL graphics
- `elm-3d-camera` provides functions for working with virtual 3D cameras inside a 3D scene
- `elm-units` provides the ability to express and manipulate scientific units
- `elm-geometry` provides the ability to manipulate 3D and 2D geometric objects
- `elm-physics` allows the simulation of real-time 3D and 2D physics scenes with rigid bodies

3D Physics Slot

The 3D Physics slot on our online code compilation system allows students to create 3D objects and apply forces to them, simulating linear and angular acceleration and collisions. This was made possible in large part by the `elm-3d-scene` and `elm-physics` packages.

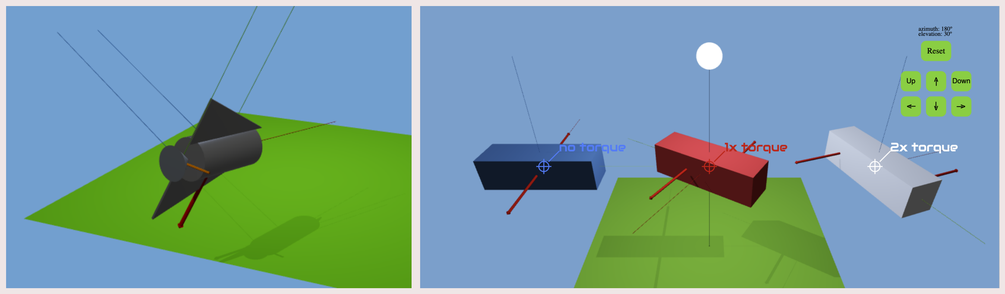


Figure 2: In our Physics Slot, students are free to explore and get familiar with 3D forces and collisions. Students can visualize the forces (red arrows) and see a realtime simulation of movements. Left: visualizing forces on a rocket as an example. Right: An example of how placing forces further from the centre of gravity causes more torque and thus a higher angular acceleration.

3D Space Simulator Game

To plan a more advanced 3D camp, students were surveyed about the type of game they would like to make next. The idea of a space game was the highest-voted activity. Thus, our advanced 3D camp had campers create and simulate 3D spaceships, complete with accurate simulated physics. A scene was set up to simulate flying from a space station to a satellite on a repair mission.

Spaceship API

To create a spaceship, students used skills acquired at the Bee Camp and special physics skills in the spaceship camp. Using an Elm API, students specify the parts of their spaceship, including fuel tanks, boosters and angular velocity stabilizers. Then they can also program keyboard actions and autopilot routines to fly their spacecraft manually, semi-automatically or fully automatically.

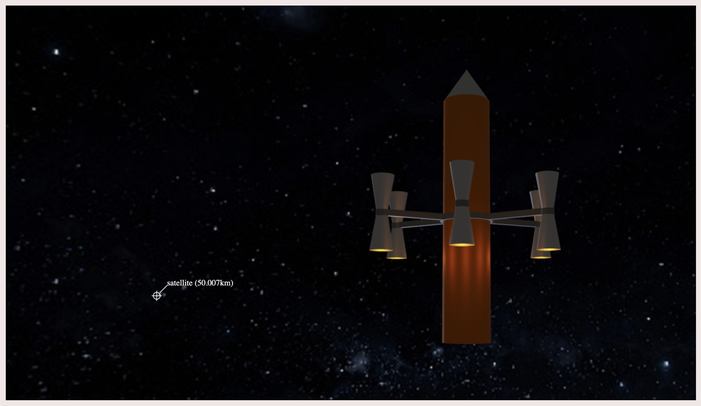


Figure 3: Students write code to design and build a rocket, which can then be flown manually or by a student-specified autopilot program. They must iterate on their design to reach their goal and optimize the amount of time it takes.

Conclusions & Future Work

The 3D camps were anecdotally well-received by campers, who found the transition from 2D to 3D challenging but manageable thanks to our familiar API. Future work includes testing students' knowledge of general and rocket physics after completing the activities, as well as studies measuring students' visualization skills after creating the 3D games. We would also like to open source the API.

Thanks

Thanks to the McMaster Dean of Engineering and NSERC for funding. Thanks to all who gave help developing these tools, and to the many volunteers, co-op students and graduate students who helped make the camps a success. Special thanks to package author and Elm community member Ian Mackenzie for providing guidance and support while developing our tools using his libraries. Most of all, thanks to the campers for your endless passion, creativity and imagination.