Helicopter Rescue 3, Cycle 2

Michael Orzel, Nicholas VonDollen, Taylor Cook, William Blair

Administrative Digest

SCHEDULES

Weekly meetings

Unlike last cycle, our group did not follow strict weekly Friday meetups. Instead, our in class meetings were used as Friday meetings were - for code review and help - as well as regrouping for the next week. Additional meetups as necessary were scheduled, mainly for presentation preparation.

FORMAL PRESENTATIONS

See testing and final presentation powerpoints in this current directory

STATUS REPORTS

None this cycle

MEMORANDA

Our goal for this cycle was to add in the features missing from the previous cycle for a minimally viable product. These included:

- Victim Rescue
    - The helicopter should have the ability to pick up and rescue victims
- Collision
    - The helicopter should be able to collide with in level objects and lose lives

Both of these features were planned for the first cycle but never completed. We were able to integrate and test these features, plus others; including music, score, start/end screens, and a pause menu. Besides these, additionally our game was redesigned with a more pleasing look and feel for the player.

Our group's progress this cycle definitely satisfied our members.

LESSONS LEARNED

Pros:
Collaborative code: like last cycle, if one member had an issue compiling, another member would run their code for them and provide feedback - this is a good example of how teamwork can benefit the group as a whole.

Cons:
Coding standards: Our group learned that strict coding practices are necessary to avoid 'spaghetti code' - although tested and working, our code is difficult to read and unorganized. As our amount of code grew larger with each feature added, it became more difficult to look through and point out relevant areas of code. We believe this could have been avoided with better coding standards; such as declaring variables at the beginning of functions, splitting up relevant code into functions, etcetera.

Helicopter Rescue 3, Cycle 2

Michael Orzel, Nicholas VonDollen, Taylor Cook, William Blair

Software Development

VERSION DESCRIPTION

Helicopter Rescue version 1.1

Description:
Helicopter Rescue version 1.1 adds the features missing from our previous version to compose a minimally viable product based on our system intent. Features added include:

-Building Collision
-Victim Rescue
-Score
-Lives
-Music
-Start/Pause/End screens

In addition, graphically the game has been redesigned with new buildings, victims, and background for a much nicer look.

The executable must now be run with the '-f *assembly_file*' option, in order to specify an assembly file for miniat to run. In addition, the -s option has been added to skip the start screen if desired.

Like the last version, the player still programs the helicopter to avoid buildings and rescue victims. All peripherals have now been implemented for the programmer to access, which include the following:

P_THRUST	- The thrust of the helicopter, -100 to 100 - read and writable
P_Y	- The y position of the helicopter (its top left) - readable
P_Y_VELOCITY	- The current speed of the helicopter - readable
P_LIVES	- The number of lives the player has - from 5 to 0
P_VICTIM_DIST	- The X distance between the front of the helicopter and the closest victim

P_VICTIM_Y - The Y distance between the helictoper and the closest victim - if the victim is above the helicopter the top of the helictoper is used, otherwise the bottom of the helicopter is used

P_GROUND_DIST - The X distance between the front of the helicopter and the closest ground building - if there isn't one currently in the helicopter's path its value is -1

P_GROUND_HEIGHT - The Y distance between the bottom of the helicopter and the top of the closest ground building

P_CEIL_DIST - The X distance between the front of the helicopter and the closest ceiling building - if there isn't one currently in the helicopter's path its value is -1

P_CEIL_HEIGHT - The Y distance between the top of the helicopter and the bottom of the closest ceiling building

There is currently only 1 level, with randomly generated buildings. The game ends when either the helicopter loses all of its lives or the end of the level is reached.

Missing:

None

Bugs:

none known currently

FEATURES

MVPP Feature #3
Name: Helicopter movement
Feature: The programmer can apply ‘thrust’ to the helicopter in the cardinal directions (up, down, left, and right). The thrust has both separate vertical and horizontal values, with a range of -100 (for backwards) and 100 (for forwards). The helicopter will not move instantaneously or stop instantaneously as well. It will have inertia and momentum adding both realism and an extra challenge for the player.
Constraints: The helicopter can move up or down on the screen but no portion of the helicopter can move offscreen. If the helicopter hits an object the player will either lose a life or lose the level. The helicopter hitting a victim results in either a save or kill, depending on velocity, as well as adding or subtracting to the player’s points.
//
MVPP Feature #6
Name: Background scrolling
Feature: The background automatically scrolls independent of the helicopter.
Constraints: The background doesn’t scroll backwards. The helicopter cannot go off screen. Scroll speed is capped so it moves slowly.
//
MVPP Feature #8
Name: Victim Generation
Feature: Victims spawn throughout the entire map for the player to rescue using the helicopter’s movement feature illustrated in Feature #3.
Constraints: The victims should spawn between a level of 5000 x 768 resolution, and should not clip through the floor or the ceiling of the level, but may clip from the front or to the back, ensuring that victims spawn within reach of the player, while still providing the rush of a challenge. The victims should scroll from right to left throughout the level so that the player-controlled helicopter may miss them.
//
MVPP Feature #9
Name: Environment Generation
Feature: Generate objects for the entire level, and make the objects scroll (1 pixel per frame).
Constraints: Make sure objects are generated properly (not overlapping). Ensure objects are not scrolling too fast or too slow. Ensure helicopter has the ability to navigate through the objects (they aren't too close together). Ensure objects move at the same speed as victims.
//

MVPP Feature #10
Name: Victim Rescue
Feature: The helicopter picks up and rescues victims; when a victim is rescued the number of saved victims increases as well as the score
Constraints: Victim can be rescued from any angle or velocity. Saving a victim adds 50 points to the score.
//
MVPP Feature #11
Name: Lives
Feature: The player starts the level with 5 lives, which are decreased by 1 each time the helicopter collides with a building. When the player has < 0 lives (-1) the end screen is displayed for losing.
Constraints: The player cannot have more than 5 lives or less than 0. If less than 0 occurs, the end screen for losing is displayed. The player cannot lose lives within the grace period given after hitting a building (invincibility frames).
//
MVPP Feature #12
Name: Title Screen
Feature: Displays the title of the game and has a start/quit button.
Constraints: The game doesn't start until the user presses the start button. In addition an 's' shortcut will exit the screen. Caps Lock cannot be on -- input only works off of lower-case "s" key. The title screen can also be skipped before startup if the user desires.
//
MVPP Feature #13
Name: Pause
Feature: Darkens the screen and halts the game, saving the current state; and has a continue button and a quit button
Constraints: The game doesn't continue until the continue button is pressed, and the current game isn't affected by pausing
//
MVPP Feature #14
Name: End Screen
Feature: Displays a graphical end screen and has a restart/quit button
Constraints: The game doesn't restart until the button is pressed, as well as quit
//
MVPP Feature #15
Name: Victim Distance Sensor
Feature: MiniAT has ports to return the X and Y distance from the helicopter to the nearest victim. The X distance is the distance between the front of the helicopter and the left side of the victim. The Y distance is the distance between the top of the helicopter and the helicopter and the bottom of the nearest victim.

Constraints: The victim which is found is the closest to the helicopter; the victim isn't behind the helicopter (already passed by);

//

MVPP Feature #16

Name: Building Distance Sensor

Feature: MiniAT has ports to return the X and Y distance to the top of both ground and ceiling buildings. The X distance for each type is the distance between the front of the helicopter and the left side of the closest ground/ceiling building. The Y distance for ground buildings is the distance between the bottom of the helicopter and the top of the building; the Y distance for ceiling buildings is the distance between the top of the helicopter and the bottom of the ceiling buildings.

Constraints: Buildings must be in front of or currently in line with the helicopter; ground and ceiling sensors are separate entities.

//

MVPP Feature #17

Name: Lose Screen

Feature: After losing all lives, the lose screen pops up, displaying a message indicating that the player has lost the game. The screen offers buttons to both restart and quit.

Constraints: Lose screen must only pop up when the player loses all lives, indicating that they have lost the game; restart starts the simulation from the beginning of the level and score/lives reset.

DESIGN

- Score:
    - Increase for each victim rescued
    - Increase for each distance checkpoint
        - The farther the checkpoint, the more it's worth
    - The faster the level is completed, the more points awarded
    - Decrease for each victim killed

- Collision:
    - Helicopter collides with obstacle: lose a life
    - Helicopter collides with a victim: if hit with helicopter, victim is killed
    - Helicopter collides with ceiling/floor of window: no problem as long as no obstacle is there
    - If collision with non-window-border environment occurs, no lives lost.
    - If collision with buildings happens at 0 lives → "You Lose" end screen plays.

- Obstacles:
    - Obstacles are part of the level, levels are premade

- Rescuing Victims:
    - The helicopter drops a rope to reach the victim
    - The victim has to ‘climb’ the rope
    - While the victim is climbing the rope they are vulnerable to other obstacles

- Helicopter Physics:
    - The helicopter follows a set of physics rules (inertia/momentum), so it won’t start or stop moving instantly

- Helicopter Movement:
    - The helicopter will have a x and y velocity, which will be affected by thrust
    - The helicopter can move anywhere on the screen
    - Cannot move backwards through the level - Super Mario Bros level progression
    - The helicopter can touch each edge of the screen and not crash, as long as an object isn’t there

- MiniAT Movement:
    - Peripherals to apply horizontal and vertical thrust, range [-100, 100]
    - Peripherals to return the helicopter’s x and y position (in exact pixels)

- MiniAT Sensors:
    - Life sensor
        - Peripheral to return the number of lives the player has left
    - Victim sensor
        - Peripheral to return the horizontal distance to the nearest victim
        - Peripheral to return the vertical distance to the nearest victim
    - Object sensor
        - Peripheral to return the horizontal distance to the nearest object
        - Peripheral to return the vertical distance to the top of the nearest object

<u>SCREENS:</u>

- Title Screen
    - Shows the name of the game, buildings, and has buttons to start or quit
- Pause Screen
    - Darkens the current play screen
    - Displays pause message
    - Has buttons for continue or quit
- End Screen
    - Has buttons for restart or quit
    - Shows credits for authors of program (Our names)
- Lose Screen
    - Displays message indicating that you lost the game.
    - Has buttons to restart game or quit.

- Has background image of cityscape.

TEST MATERIAL

See spreadsheet in current directory

SOURCE CODE

See source code in parent directory