Coding 2, Weekly Homework

OpenFramework

Week 2

GitHub Link https://github.com/Pannic17/C2-Week02

My Week 2 homework consist of 2 parts:

Week02O, Lines
https://ual.cloud.panopto.eu/Panopto/Pages/Viewer.aspx?id=124181b4-1dd9-4859-90ae-afc501287e6c
Week02O, Lines, is a line-particle system with randomly generated particles of circle with a tail of lines.

Week02V, Heart
https://ual.cloud.panopto.eu/Panopto/Pages/Viewer.aspx?id=2d6de9bc-49f2-4e18-a1b3-afc501287e70
Week02V, Heart, is a particle system consists of red circles that forms a bumping heart.

Week 3

GitHub Link https://github.com/Pannic17/C2-Week03
https://github.com/Pannic17/C2-Week03-Boids

As Week 3 can be done in group, I cooperated with two of my classmates: Zhou Fang & Yiren Wang.

With Zhou Fang, Lead Goose
https://ual.cloud.panopto.eu/Panopto/Pages/Viewer.aspx?id=f47d3cd3-f907-4b52-b243-afc60076e06f
Lead Goose extract and remodified some of the functions and parameters of *Boid* class and create a parent class called *Creature*. Then I create a child class called *LeadGoose* of *Creature* that behave differently from *Boid*. The idea is that boids are the birds that follows the lead goose around the screen. There is only one *LeadGoose* which follows the click of mouse when user clicks one the screen. To make the boids more adherence to the lead goose, *Boid* class will auto calculate their distance to the lead goose and apply vector velocity towards the lead goose.

With Yiren Wang, *Runaway*, Bubbles with Tails

https://ual.cloud.panopto.eu/Panopto/Pages/Viewer.aspx?id=8761debd-6ad4-4d16-92f9-afc6003c62f3

Runaway create a scene that fulfills bubbles with tails. The idea is to a child class called *BoidLife* that record and forms the a tail for the initial *Boid*. To realize it, the tail particle only have to stay at the position where boid had past and keep the velocity of the boid at that time, basically is the replica of the boid. However, the tail (*BoidLife*) would have to disappear after certain time and I want to add a size decay for the tail, so that the tail would be more beautiful, which differ their behavior from the *Boid* class.

Python

Week 5

GitHub Link https://github.com/Pannic17/C2-Week05

Video Link https://ual.cloud.panopto.eu/Panopto/Pages/Viewer.aspx?id=b1b91f0b-ede1-40a0-967e-afc6003bc661

For the python riddle, I tried to hide information inside the images using OpenCV library. OpenCV has built-in image hash function, which generates list of image hash in many different types. The hash is binary and can be converted to hexadecimal, so it can be used as coordinates or RGB value.

I implemented 2 types of hiding information.

hash_hide

Resizes the image to 256x256 pixels.
Hashes the image using MarrHildrethHash.
Converts the hash values to a hexadecimal representation.
Use hash values to generate a list of coordinates represent the pixels to be modified.
Replaces pixels in the image with ASCII characters based on the hash values.
Displays the modified image using the OpenCV imshow function.

hash_glitch

Resizes the image to 256x256 pixels.
Hashes the image using MarrHildrethHash and BlockMeanHash.
Converts the hash values to a hexadecimal representation.
Converts the hash values to RGB values.
Adjusts the y-coordinate of pixels in the image based on the hash values.
Displays the modified image using the OpenCV imshow function.

Week 6

GitHub Link https://github.com/Pannic17/C2-Week06

Video Link https://ual.cloud.panopto.eu/Panopto/Pages/Viewer.aspx?id=dbbe0e03-67fe-449f-82fb-afc60031f775

For the image transformation, I collected a dataset of approximately 600 cat images, then I normalized the image and use *Cascade Classifier* in *OpenCV* to process the

image, which will detect the cat faces and returns the coordinate of the cat face. Then I use the coordinate to crop the image, thus, I will get a dataset of cat faces. The *Cascade Classifier* is not so accurate, to avoid faulty result, I filtered the image to abort any cat faces that is smaller than 10x10 pixels, and also enlarge the cat's face a little bit.