**PMC-lab 5**

**A.**

**\*Write up- Can you think of ways to improve this?**

I made this with infinite loop so that flip book continually moves.

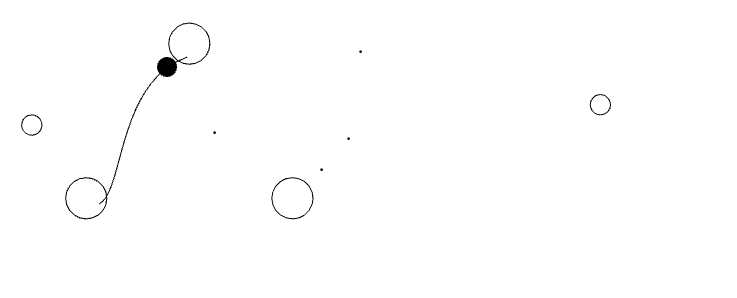
**B.**

**\*Write-up:**Is the movement smooth and could you think of ways to improve this?

Yes it’s smooth.

**C.**

1. Make a copy of your code from part B.  Modify it to change the *path* the object travels to follow a curve. To do so, you will need to define two arbitrary control points for each in-between section. How does the curve change your perception of the movement?



1. Start from another copy of the part B code. This time, modify it so that the *path* is still linear, but  *distance* that is traveled between frames is controlled by a spline. Do do this, your code needs to take the current frame number and input it into a curve() function. The output of that will be the input to your lerp() function, which will give you the location of the moving object.



The distance between the two keyframes are controlled by spline while the path moving between the two key frames are still linear.

**PMC-lab 6 Physical Simulation**

**A.**

**\* Write-up:**What happens to the velocity of the object? How about the acceleration?

I think the circle is being accelerated slightly. So, velocity is increasing.

\*change the code so that the ball always start from the middle of canvas

location.set(ofGetWidth()/2, ofGetHeight()/2);

\*Try different values of the velocity and acceleration so that it looks like a cannon ball shooting horizontally or slightly upwards with a 45 degree angle

🡪You need to press space bar to shoot the cannon ball

velocity.set(10,-10);

acceleration.set(0, 0.58);

I just set the velocity and acceleration like above. It draws a cannon ball that slightly upwards with a 45 degree at first and downwards after it reaches the highest point.

\*Now add some interactive features: pressing the space bar to shoot a new cannon ball, and using the position of the mouse to change the initial velocity

🡪when you press space bar, you can shoot cannon ball. And depends on where your mouse is, you can decide velocity and direction of your cannon ball.

//in dot::move() file

velocity.set(ofGetMouseX(),ofGetMouseY()); //instead of set the velocity in dot::dot()

//in ofApp.h file

int result;

//in ofApp.cpp file draw function

if (result) {

myDot.run();

}

//in ofApp.cpp file keyPressed function

if (key == ' ') {

result = 1;

}

\* Can you think about other ways to make it more interactive and interesting?

I tried to set initial position of a cannon ball under the control of a user.

First of all, I made getP function in public part of the dot class to access to location in the ofApp.cpp file which is private variable

void getP();

And then, I declared getP function in dot.cpp file

void dot::getP()

{

location.set(ofGetMouseX(),ofGetMouseY());

}

Finally, I made getP function called when user press ‘t’. It gets and fixes the x and y position.

//in keyPressed function

if (key == 't') {

result = 2;

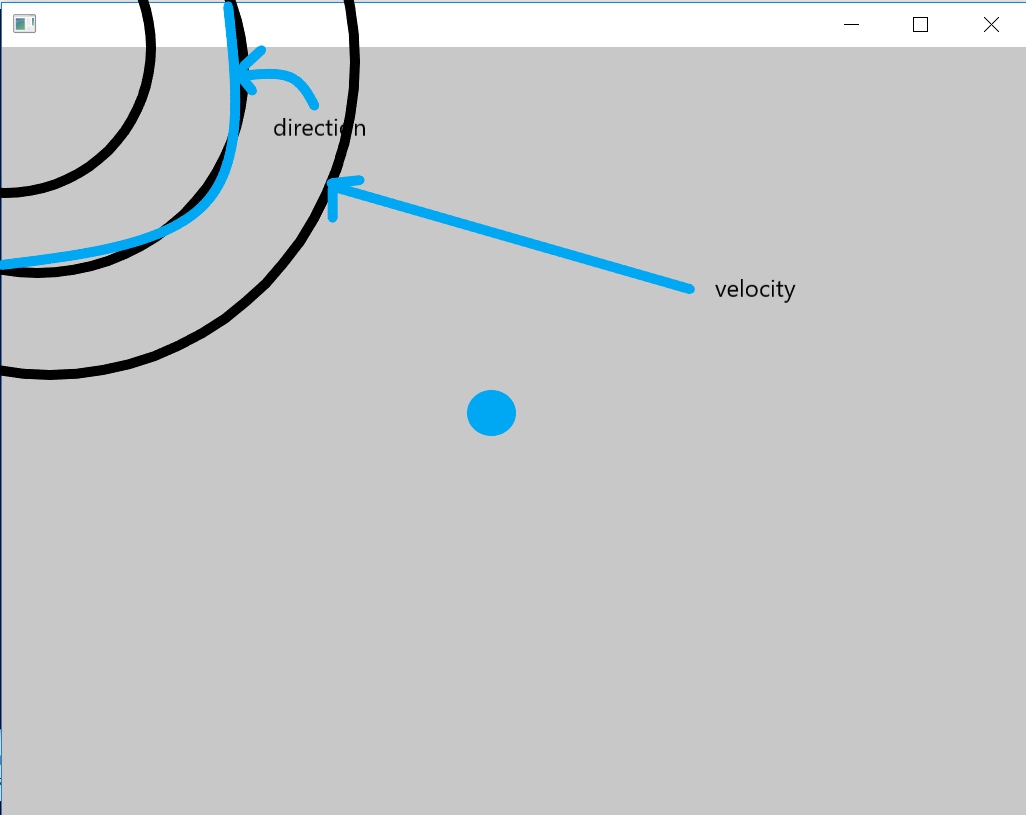
}

//in draw function

if (result == 2) {

myDot.getP();

}



**B. Springs**

**\***When you change the parameters (M, K, D, R), how does the animation change?

float M = 0.8; // Mass

float K = 0.2; // Spring constant

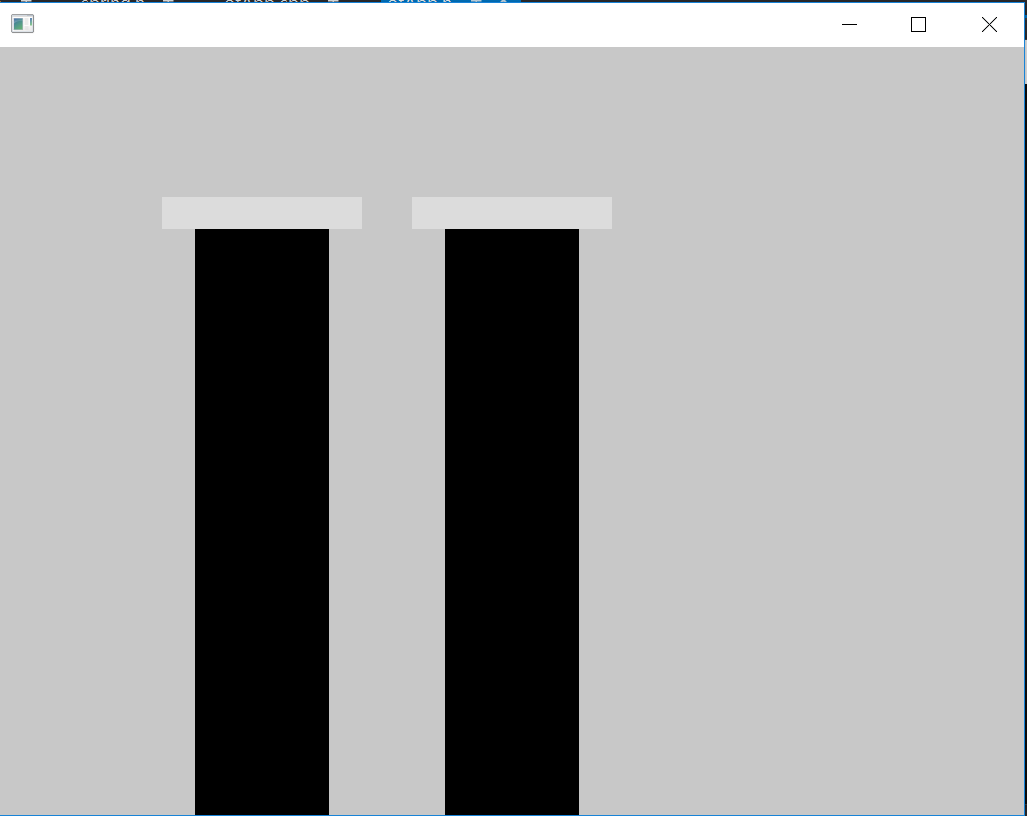
float D = 0.92; // Damping

float R = 150; // Rest position

M which stands for mass decides the spring’s weight. K stands for spring constant, D stands for damping, and R decides rest position. Since K is a constant factor, it refers to characteristic of the spring such as stiffness. Each spring has their own K value. When the force is F, F=K\*x is valid, where x scales linearly with respect to the distance. Damping force is introduced to reduce the motion of an object like friction.

\*Add more springs with different parameters to your code so you can compare them side by side - you should first change the code to create a spring class

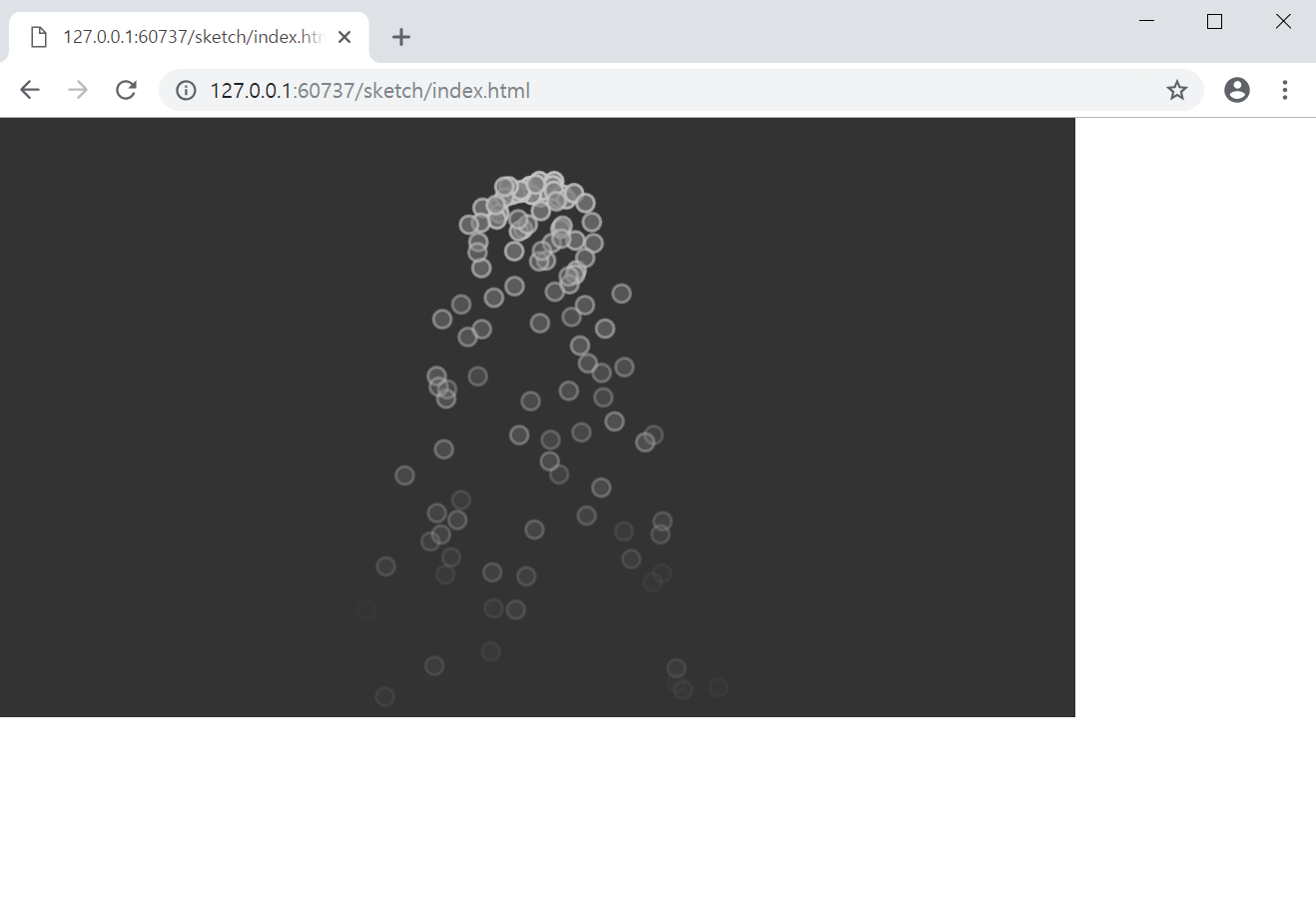
First of all, I changed the code to create a spring class. I added spring.h and spring.cpp file, and I changed some code to enable to draw multiple springs at different locations.



I made the left spring lighter of the left spring for comparison. Looking at the left spring, It comes back more quickly when I push in it a downward direction. On the other hand, the right spring comes back more slowly when I push it in the same direction.

\*Can you think about more creative ways of using the springs to generate some interesting physical simulation effects?

**C. Particle systems**

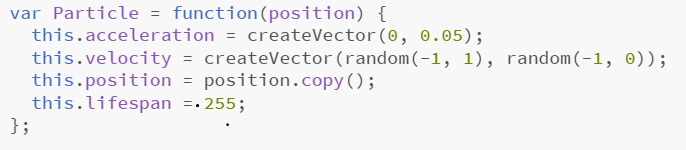


**\*Write-up:** *How are these five steps implemented in this example?*

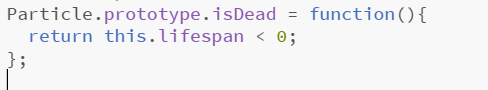
For each frame, first, a new particle that are born during this frame are generated.



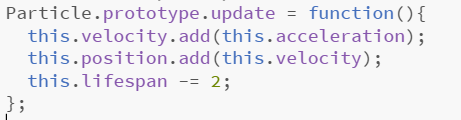
Second, the particle system is assigned attributes such as position, velocity, size, colour and lifetime.



Third, the particle is deleted if the particle has exceeded its lifetime.

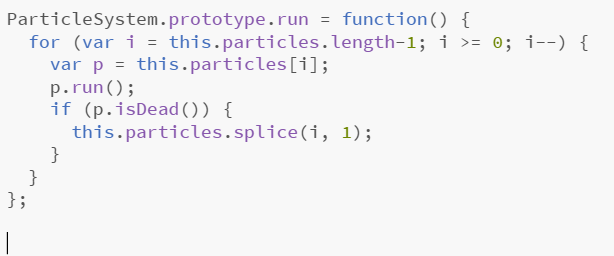


Fourth, the particle is animated and their shading parameters changed according to the controlling processes.



The above part gives animation by updating velocity, position, and lifespan.

Fifth, the particle is rendered.



**PMC-lab7**

\***Rotate()**(or **ofRotate()**) the triangle to make it align with its velocity using either: **angleBetween()** (p5.js, [reference](https://p5js.org/reference/#/p5.Vector/angleBetween)) or **angle()** (ofx, [reference](https://openframeworks.cc/documentation/math/ofVec2f/#show_angle)).

After I change the velocity, I used ofRotate() to align the tip of the triangle with the direction of its velocity.

//Add this code in ofApp.h file

float angle;

ofVec2f velocity2;

velocity2.set(0, -1);

velocity.set(0.3,0.4);

angle = velocity2.angle(velocity);

ofRotate(angle);

\***Write-up:** Does your code handle the situation where velocity is (-0.3, -0.4)? (The answer is different in p5.js and openFrameworks)  If not, why? How could you fix it? (hint: use dot product)

In openframeworks, velocity(-0.3,-0.4) is handled. If I assume that it does not work, we need to rotate by using ofRotate(), and dot(). First, we need to get angular relationship between v1(0.3,0.4) and v2(-0.3, -0.4) by using dot(). And put the value in dot. And rotate the triangle by ‘dot’ degree.

\*Modify your code to calculate the rotation using dot product?

//in ofApp.h file

ofVec2f v1;

ofVec2f v2;

float dot; //a value of angular relationship

float mag1; //magnitude1

float mag2; //magnitude2

float cos; //cosine

float theta; //angle

//in ofApp.cpp file setup function

v1.set(0.3, 0.4);

v2.set(-0.3, -0.4);

dot = v1.dot(v2); //-0.25

mag1 = v1.length(); //0.5

mag2 = v2.length(); //0.5

cos = dot / (mag1\*mag2);

theta = acos(cos);

//in draw function

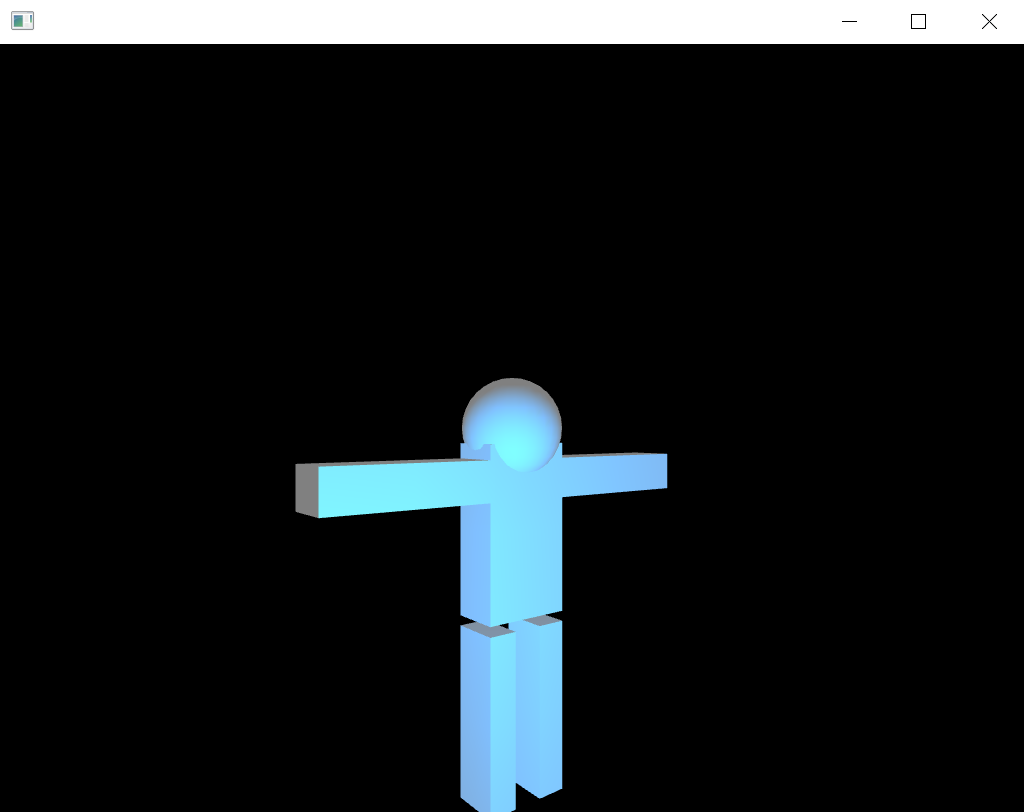
ofRotate(angle);

ofRotate(theta / PI \* 180);

This is the code that I tried to get theta value of v1 and v2 by using acos(), length(), and dot() instead of angle(). And theta needs to be changed to degree, so I put theta/PI\*180 as a parameter of ofRotate()

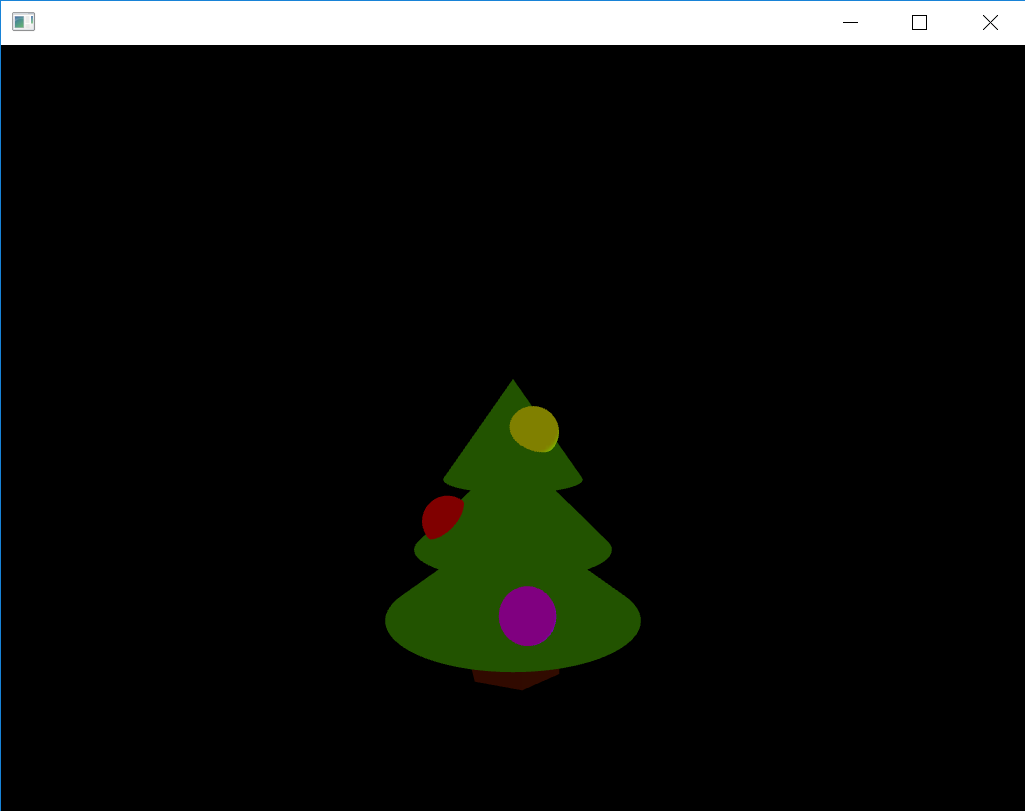
**B.**

**\***Try to make a 3D stick figure with box and sphere



**\***Can you try to draw something more interesting than a stick figure?

I tried to draw a Christmas tree.



**C.**

**\***Can you visualise the cross product of two vectors as shown in Figure B?

**//in ofApp.h file**

ofEasyCam cam;

ofVec3f a;

ofVec3f b;

ofVec3f c; //cross product

**//in ofApp.cpp file setup function**

a.set(1, 0, 0);

b.set(0, 1, 0);

c = a.getCrossed(b);

**//in ofApp.cpp file draw function**

ofBackground(20);

ofTranslate(ofGetWidth() / 2, ofGetHeight() / 2, 0);

ofSetColor(255);

cam.begin();

// Red line from origin on X axis

ofSetColor(ofColor::red);

ofDrawLine(a.x, a.y, a.z, 200, 0, 0);

//Green line from origin on Y axis

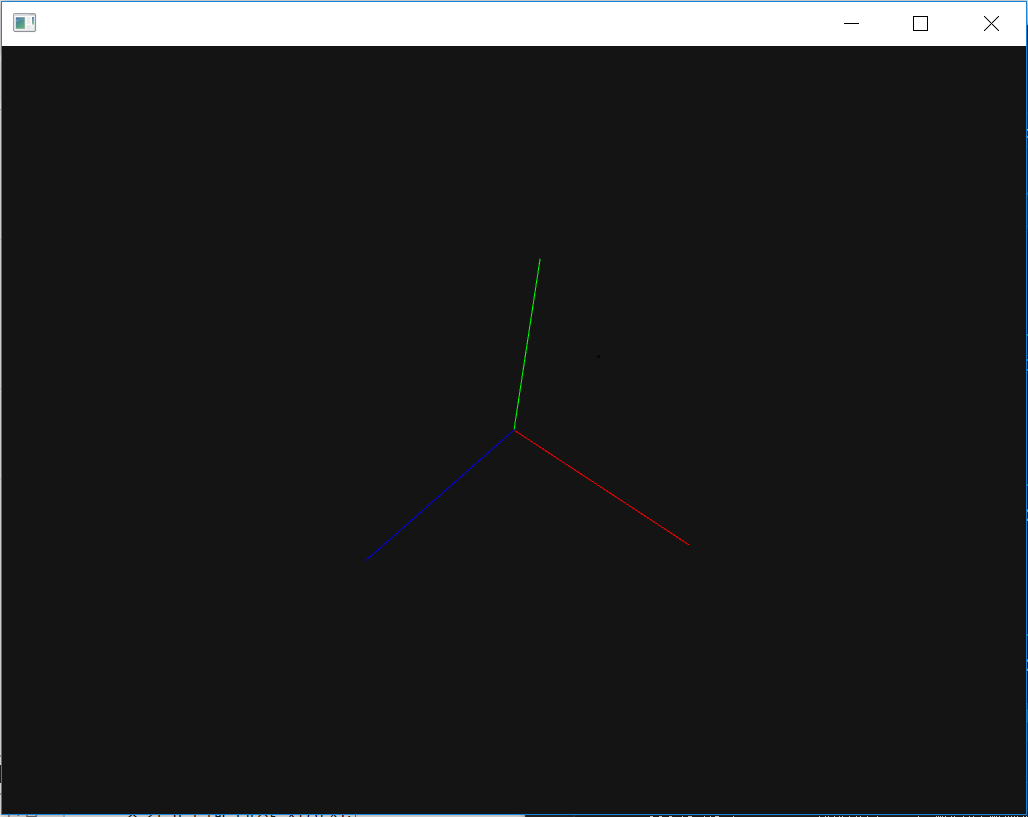
ofSetColor(ofColor::green);

ofDrawLine(b.x, b.y, b.z, 0, 200, 0);

//Blue line from origin on cross product of the two vectors

ofSetColor(ofColor::blue);

ofDrawLine(c.x, c.y, c.z, 0, 0, 200);



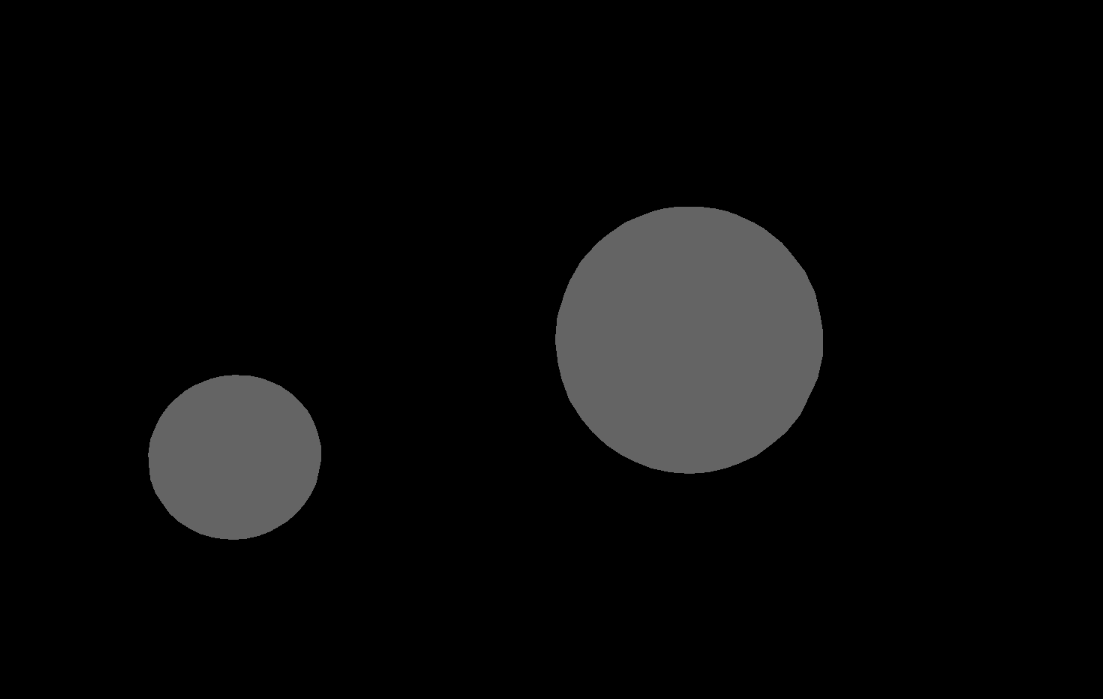
**\***What’s the difference between b.cross(a) and a.cross(b)?

The cross product does not follow the commutative property because the direction of the unit vector becomes opposite when the vector product occurs in a reverse manner. So, b x a is equal to – (a x b). So, b.cross(a) means a x b while a.cross(b) means a x b. And These two values are additive inverse of each other.

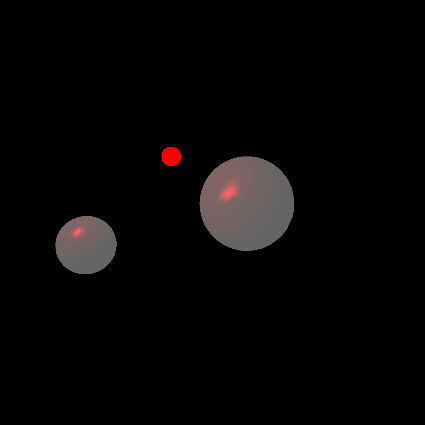
**D.**

\*try putting some comments in the code

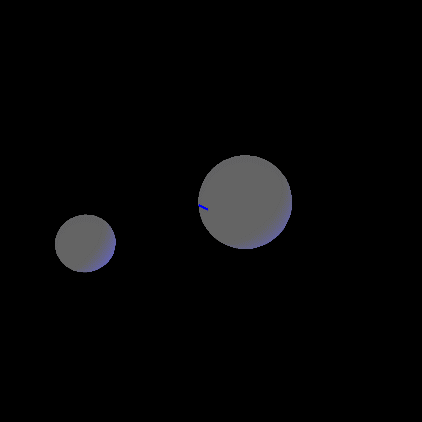
\*Try to switch off one or two of the lights and observe the three lights separately. **Write-up:** What do you see?

// only when ambient light is on

I can see the two sphere primitives when the ambient light is on.

// when ambient light and pointLight are on

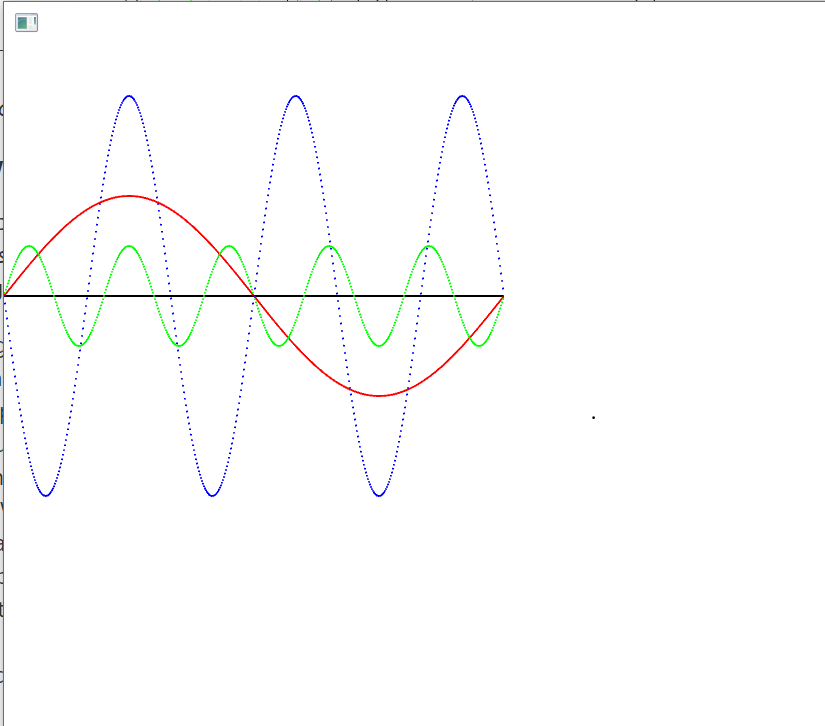
I can see the specific point of the light origin. I can see that light reflects differently when the light is farther from or nearer to the object.

// when ambient light and directionalLight are on

I can see the direction of the light, but it cannot be positioned closer or farther away from a geometry since they do not have a specific point of origin.

**PMC\_lab8\_Sound**

**A. Understanding the sine equation**



**B. Experimenting with sound**

1. What is the highest frequency you can hear?

When I keep amplitude of the sine wave approximately 0.01, the highest frequency that I can hear is about 7200Hz.

1. Does this change when you are using headphones versus the built-in computer speakers?

In terms of the built-in computer speaker, I can hear up to about 6050Hz.

1. When you keep amplitude of the sine wave constant, what is the loudest frequency for you when you are using headphones?

When frequency is about 4300Hz, sound is the loudest for me.

1. What is the loudest frequency when you are using the built-in computer speakers? Is it the same as when you are using headphones? Why do you think this is the case?

When I’m using the built-in computer speaker, about 3700Hz is the loudest frequency for me. I think it’s because, when I use headphone, I can hear the sound clearly without other sound pressures compared to the built-in computer speaker. For more explanation, Volume is affected by sound pressure level. A sound wave in a transmission medium causes a deviation in the local ambient pressure, static pressure.

Sound pressure, denoted *p*, is defined by

Where is the total pressure, and is the static pressure.

Sound pressure level, denoted and measured in dB, is defined by

Where is the root mean square sound pressure, is the reference sound pressure, and is the neper.

So, in the case of hearing with built-in computer speakers, had been added as an ambient pressure, so that is slightly decreased. However, in terms of hearing with headphones, headphones blocks ambient sound so that that I can hear can increase.

1. Using the SineWaveKeyboard example, listen to the difference between a sine at 400 Hz and a sine at 500Hz. (In music, this difference is called an “interval.”) Now, listen to the difference between a sine at 1600 Hz and a sine at 1700 Hz. Does the 1600/1700Hz pair sound closer together or further apart than the 400/500Hz pair?

1600/1700Hz pair sound is even closer together than 400/500Hz pair.

1. 500 Hz is exactly 100 Hz above 400 Hz, and 1700 Hz is exactly 100 Hz above 1600 Hz. So why don’t these two intervals sound the same?

Because, human hear notes separated by octaves. Octave is the interval between one musical pitch and another with double its frequency. And octave is separated by 12 notes, and human sounds note with the same letter name similar which is called tine chroma. So, octaves increasing, frequencies between adjacent two notes increase. Even thought it’s the same 100Hz interval, it sounds different depends on what octave they are in for a human ear.

1. What frequency would you pair with 1600Hz to get the same interval (perceptual distance) as the 400/500Hz pair?

To get the same interval as the 400/500Hz pair, it would be 2000Hz.

**C. Adding sines together**

1.

* What happens to the pitch of the sound as you add more frequencies?

Pitch is harmonically balanced stabilized. The pitch is getting higher entirely as I add more higher frequencies.

* What happens to the sound colour / timbre / tone quality as you add more frequencies?

Tone quality is same. There is no difference because those sounds are from same instruments. I can’t distinguish one frequency from others.

\*\*When you test my code in the file, please drag the slider on the top of the screen, then another slider will be shown. And ‘q’, ‘w’, and ‘e’ is the buttons that you can choose sound would like to hear.

2.

* What do you hear as you add more frequencies? How is this different from when you added frequencies that were harmonically related?

When I add more frequencies that aren’t harmonically related, I can hear the various frequencies at the same time depends on what frequency and amplitude are added. And also I can make any harmony. On the other hand, I could only hear that pitch became high and harmony is same when I add harmonically related frequencies.

* How does the sound change when the range of frequencies is very big (e.g., a spread of thousands of Hertz) or very small (e.g., all within a few Hertz of each other)?

When the range of frequencies is very small, it’s hard to distinguish each sound. However, it’s easy to distinguish each sound when the range of frequencies is very big.

* For extra fun, try using an array of **maxiOsc** objects so that you can easily add many (dozens!) of sines together at random frequencies. This will create some interesting audio effects…

**PMC\_lab9\_Sound2**

**A. Additive synthesis**

**a. Writeup**: Recall that the generic equation for a sine wave at time t is:

f(t)=Asin(2πft+ϕ)

Let's assume that the value of f is 100 Hz.

1. What is the frequency of the first sine wave in the square wave summation (i.e., the sine wave that corresponds to k=1 in the top equation)?

The frequency of the first sine wave in the square wave summation is 100Hz

1. What is the amplitude of the first sine wave?
2. What is the phase of the first sine wave?

0

1. What the frequency of the second sine wave (where  k=2)?

300Hz

1. What is the amplitude of the second sine wave?
2. What is the phase of the second sine wave?

0

1. What is the frequency of the k-th sine wave? (i.e., express the frequency as a function of k)
2. What is the amplitude of the k-th sine wave? (i.e., express the amplitude as a function of k)
3. What  is the phase of the k-th sine wave?

0

**b.** Now edit the q2Additive\_skeleton sketch ([download it from learn.gold](https://learn.gold.ac.uk/mod/resource/view.php?id=623052)) to implement the creation of a square wave using a finite number of sine waves. Specifically, fill in the for-loop in the **setup()** function so that the first numHarmonics sines are generated and played. (Note that playing them simultaneously is the same thing as adding them together, in terms of the waveform output by your computer.) Then, answer the following questions:

\*This is where I’ve changed. (also numHArmonics and fundamental where needed)

myFreqs[k] = 100\*(2\*k-1); //change this

myAmp[k] = (4/TWO\_PI)\*(1/(2\*k-1)); // and this

1. What does the waveform look like when numHarmonics = 1?

It looks like normal sine wave where k=1, f=100, a=0.6366197723675814.

1. How does the waveform shape change as the number of harmonics increases?

It gradually becomes square wave

1. How does the sound change as the number of harmonics increases?

Tone colour, amplitudes, and frequencies change as the number of harmonics increases.

1. How does the sound change as the fundamental frequency increases?

Then, volume and pitch change.

1. What happens to the sound when you use a fundamental frequency of around 1000Hz

and a very large number of harmonics (e.g., 500)?

When it has a very large number of harmonics, it becomes quite accurate square wave. And when it use a fundamental frequency of around 1000Hz, the interval of the wave will be smaller than when it use that of 400Hz.

**B.** **Intro to reverb using convolution**

1. Try different impulse responses. How does the resulting sound change?

I recorded short humming and clap sound and added it with a set of pre-recorded impulse response. In result, it changed my voice and clap sound. It convolved my voice so that it sounds like the sound took place in the same acoustic environment where the impulse response was recorded.

**C. Aliasing**

**Write-up**: Use the sineKeyboard and/or sineMouse sketches to explore the following questions.

1. These sketches run with a sampling rate of 44100Hz. What is the Nyquist frequency?

22050Hz

1. What frequency will you hear if you synthesise a sine wave at 10,000Hz? Why?
2. What frequency will you hear if you synthesise a sine wave at 34,100Hz? Why?
3. What frequency will you hear if you synthesise a sine wave at 43,100 Hz? Why?
4. Edit the SineWaveMouse sketch so that the horizontal mouse position is mapped from 0Hz at the left-most edge of the screen to 100,000Hz at the right-most edge of the screen. You’ll need to edit the line:   **freq=map(mouseX, 0, width, 20.0, 20000.0);**

freq = map(mouseX,0,width,0, 100000);

1. What do you expect the pitch to do as you move from 0Hz to 22,050Hz? Why? Verify that the sketch makes the sound you expect.

12 Upward pitches will be repeated(Octaves increase)

1. What do you expect the pitch to do as you move from 22,050Hz to 44,100Hz? Why? Verify that the sketch makes the sound you expect.

12 Downward pitches will be repeated. Because folding is occurred.

Pitch will be going down(Octave decrease) because of folding when the rates are above the Nyquist frequency.

1. What do you hear when you move from 44,100Hz to 88,200Hz? How does this compare to what you hear when moving from 0Hz to 44,100Hz?

Those are same when I hear those sounds.

1. What does this suggest about aliasing when your frequency is above the sample rate?

Aliasing happens when your frequency is above the half of the sampling rate.