Final Project - Jiatong Zhu - 20007095

1. Proposal：

Before taking this semester class, I didn't understand the working principle of many interactive computing art works, but now I know most of this aspect comes from the ofGUI plug-in and openCV plug-in of openframework. At the beginning of the project, I planned to do some beautiful graphical interactions. Until I saw the openframework image interaction studied by Denis Perevalov, I realized the potential of dynamic, intelligent, artistic and interactive artwork. And I started the journey of exploring openframework through the project of creating a video synthesizer, in which I learned the GUI, set up a particle system, load and draw images, play video files, grab live video from the camera, and use additive blending to mix layers , Use shaders to create video effects. Next, I plan to use GUI to mix 2d and 3d. I hope to make it mirror the real scene while maintaining a certain degree of artistic complexity.

2. Tools and frameworks used in development:

C++ , OpenFramework

Learning Materials：

* OpenFramework Website: https://ofxaddons.com/categories
* Circle loop: https://www.youtube.com/watch?v=NBpBR2gRPEE
* A book < openFrameworks Essentials >

3. Final project: Kaleidoscope ---- Record the development process

Link1: https://youtu.be/IbAtL2r9Huo

Link2: https://youtu.be/iNy22dR-LgY

Development steps:

1) Loading and drawing raster images and videos
2) Grabbing and drawing live video from a camera
3) Mixing images using additive blending
4) Drawing to the offscreen buffer
5) Processing images using fragment shaders

Challenge:

The core of my project is to create a video synthesizer. There are two difficulties that need to be resolved in the project

implementation.

A. How to process the real-time video in the camera

B. Choose a video mapping plug-in

C++ programming:

main.cpp

```
1 #include "ofMain.h"
```

```
#include "ofApp.h"

//========================================================================
int main( ){
    ofSetupOpenGL(1024,768,OF_WINDOW);                    // <-----

    // this kicks off the running of my app
    // can be OF_WINDOW or OF_FULLSCREEN
    // pass in width and height too:
    ofRunApp(new ofApp());

}
```

ofApp.cpp

```
#include "ofApp.h"

//--------------------------------------------------------------
void ofApp::setup(){
    ofSetWindowTitle( "Video synth" ); //项目的窗口标题
    ofSetWindowShape( 1280, 720 ); //屏幕大小
    ofSetFrameRate( 60 ); //渲染帧速率
    ofBackground( ofColor::white ); //背景色

    gui.setup( "Parameters", "settings.xml" ); //将标题设置为parameter
    gui.add( countX.setup( "countX", 50, 0, 200 ) );
    gui.add( stepX.setup( "stepX", 20, 0, 200 ) );
    gui.add( twistX.setup( "twistX", 5, -45, 45 ) );

    gui.add( countY.setup( "countY", 0, 0, 50) );
    gui.add( stepY.setup( "stepY", 20, 0, 200 ) );
    gui.add( twistY.setup( "twistY", 0, -30, 30 ) );
    gui.add( pinchY.setup( "pinchY", 0, 0, 1 ) );

    //设置组及控件
    globalGroup.setup( "Global" ); //将组的标题设置为“Global”
    globalGroup.add( Scale.setup( "Scale", 1, 0.0, 1 ) ); //将滑块添加
    globalGroup.add( Rotate.setup( "Rotate", 0, -180, 180 ) );
    globalGroup.add( Background.setup("Background",255,0, 255));
```

```
    gui.add( &globalGroup ); //将组添加到GUI面板

    primGroup.setup( "Primitive" );
    primGroup.add( shiftY.setup("shiftY",0.0,-1000.0,1000.0 ) );
    primGroup.add( rotate.setup("rotate",0.0,-180.0,180.0 ) );
    primGroup.add( size.setup( "size",
      ofVec2f(6,6),
      ofVec2f(0,0),
      ofVec2f(20,20) ) );
    primGroup.add( color.setup( "color",
      ofColor::black,
      ofColor(0,0,0,0),
      ofColor::white ) );
    primGroup.add( filled.setup( "filled", false ) );
    primGroup.add( type.setup( "type", false ) );
    gui.add( &primGroup );

    showGui = true;

    gui.loadFromFile( "settings.xml" ); //要gui在启动时加载状态

    ofLoadImage( image, "collage.png" );
    video.loadMovie( "flowing.mp4" );
    video.play();
    mixerGroup.add( imageAlpha.setup( "image", 100,0,255 ) );
    mixerGroup.add( videoAlpha.setup( "video", 200,0,255 ) );
    mixerGroup.add( cameraAlpha.setup( "camera", 100,0,255 ) );

    shader.load( "kaleido" );
    mixerGroup.add( kenabled.setup( "kenabled", true ) );
    mixerGroup.add( ksectors.setup( "ksectors", 10, 1, 100 ) );
    mixerGroup.add( kangle.setup( "kangle", 0, -180, 180 ) );
    mixerGroup.add( kx.setup( "kx", 0.5, 0, 1 ) );
    mixerGroup.add( ky.setup( "ky", 0.5, 0, 1 ) );

    gui.minimizeAll();
    gui.add( &mixerGroup );

    fbo.allocate( ofGetWidth(), ofGetHeight(), GL_RGB ); //设置缓冲区
```

```
64
65 }
66
67 //--------------------------------------------------------------
68 void ofApp::update(){
69
70     video.update();//定期更新视频对象
71     if ( camera.isInitialized() ) camera.update();
72
73 }
74
75 void ofApp::draw2d(){
76
77     ofDisableSmoothing();
78     ofEnableSmoothing();
79
80     ofEnableBlendMode( OF_BLENDMODE_ADD );
81     ofBackground( Background );
82     ofSetColor( 255,imageAlpha );
83     image.draw( 0, 0, ofGetWidth(), ofGetHeight() );
84     //用于绘制图像，结果将在整个屏幕上拉伸图像
85     ofSetColor( 255,imageAlpha );
86     video.draw( 0, 0, ofGetWidth(), ofGetHeight() );
87     if ( camera.isInitialized() ) {
88       ofSetColor( 255,imageAlpha );
89       camera.draw( 0, 0, ofGetWidth(), ofGetHeight() );
90     }
91     ofEnableAlphaBlending();
92 }
93
94
95 void ofApp::exit() {
96   gui.saveToFile( "settings.xml" );
97 }
98
99 void ofApp::stripePattern(){
100     //ofSetColor( ofColor::yellow );
101     ofSetColor( color );
102     //ofSetLineWidth( 3.0 );
```

```
103     ofSetLineWidth( 1 );
104     //ofNoFill();//禁用填充模式
105     if ( filled ) ofFill(); else ofNoFill();
106
107     for (int i=-countX; i<countX; i++) {
108       ofPushMatrix();
109       ofTranslate( i*stepX, 0 );
110       ofRotate( i * twistX );
111
112       //ofScale( 5, 5 );
113       //ofTriangle( 0, 0, -50, 100, 50, 100 );
114       //ofLine( 0, -100, 0, 100 );
115
116       ofTranslate( 0, shiftY );
117       ofRotate( rotate );
118       ofScale( size->x, size->y );
119       if ( type ) ofRect( -50, -50, 100, 100 );
120       else ofTriangle( 0, 0, -50, 100, 50, 100 );
121
122       ofPopMatrix();
123     }
124       ofScale( 6, 6 );
125       ofTriangle( 0, 0, -50, 100, 50, 100 );
126 }
127
128
129 void ofApp::matrixPattern() {
130   for (int y=-countY; y<=countY; y++) {
131     ofPushMatrix();
132     //---------------------
133     if ( countY > 0 ) {
134       float scl = ofMap( y, -countY, countY, 1-pinchY, 1 );
135       ofScale( scl, scl );
136     }
137     ofTranslate( 0, y * stepY );
138     ofRotate( y * twistY );
139     stripePattern();
140     //---------------------
141     ofPopMatrix();
```

```
142   }
143 }
144
145
146 //--------------------------------------------------------------
147 void ofApp::draw(){
148
149     fbo.begin();
150     draw2d();
151     fbo.end();
152
153     ofSetColor( 255 );
154
155     if ( kenabled ) {
156       shader.begin();
157       shader.setUniform1i( "ksectors", ksectors );
158       shader.setUniform1f( "kangleRad", ofDegToRad(kangle) );
159       shader.setUniform2f( "kcenter", kx*ofGetWidth(), ky*ofGetHeigh
160       shader.setUniform2f( "screenCenter", 0.5*ofGetWidth(), 0.5*ofG
161     }
162     fbo.draw( 0, 0, ofGetWidth(), ofGetHeight() );
163     if ( kenabled ) shader.end();
164     /*
165     ofPushMatrix();
166     ofTranslate( ofGetWidth() / 2, ofGetHeight() / 2 ); //将坐标系原点
167     //----
168     float Scl = pow( Scale, 4.0f );
169     ofScale( Scl, Scl );
170     ofRotate( Rotate );
171
172     //stripePattern(); //调用函数
173     matrixPattern(); //使用滑块的值
174
175     //----
176     ofPopMatrix();
177     //gui.draw();
178     */
179
180     if ( showGui ) gui.draw(); //在屏幕上绘制面板
```

```
181
182 }
183
184 //--------------------------------------------------------------
185 void ofApp::keyPressed(int key){
186     if ( key == 'z' ) showGui = !showGui; // 检查是否按下Z，使用协商运算
187     if ( key == OF_KEY_RETURN ) ofSaveScreen( "screenshot.png" );
188     if ( key == 's' ) {
189       ofFileDialogResult res;
190       res = ofSystemSaveDialog( "preset.xml", "Saving Preset" );
191       if ( res.bSuccess ) gui.saveToFile( res.filePath );
192
193       //使用系统加载对话框加载预设
194       if ( key == 'l' ) {
195         ofFileDialogResult res;
196         res = ofSystemLoadDialog( "Loading Preset" );
197         if ( res.bSuccess ) gui.loadFromFile( res.filePath );
198       }
199     }
200
201     if ( key == 'c' ) {
202       camera.setDeviceID( 0 );
203       camera.setDesiredFrameRate( 30 );
204       camera.initGrabber( 1280, 720 );
205     }
206 }
```

ofApp.h

```
1 #pragma once
2
3 #include "ofMain.h"
4 #include "ofxGui.h" // GUI插件
5
6 class ofApp : public ofBaseApp{
7     //声明可视面板和四个滑块
8     ofxPanel gui; //声明GUI面板
9     ofxIntSlider countX; //声明countX滑块，其中包含整数值
```

```
    ofxFloatSlider stepX; //step滑块
    ofxFloatSlider twistX; //twist滑块

    ofxGuiGroup globalGroup; //用于创建一组控件
    ofxFloatSlider Scale;
    ofxFloatSlider Rotate;
    ofxFloatSlider Background;

    ofxGuiGroup primGroup; //再创建一组来调整几何图元的绘制参数
    ofxFloatSlider shiftY, rotate;
    ofxVec2Slider size;
    ofxColorSlider color;
    ofxToggle filled, type;

    //矩阵模式
    ofxIntSlider countY;
    ofxFloatSlider stepY, twistY, pinchY;

    ofTexture image; //绘制图像文件
    ofVideoPlayer video; //加载视频文件
    ofVideoGrabber camera;
    ofxGuiGroup mixerGroup;
    ofxFloatSlider imageAlpha, videoAlpha, cameraAlpha;//调音台
    ofFbo fbo;//声明屏幕外缓冲区
    ofShader shader;
    ofxToggle kenabled;
    ofxIntSlider ksectors;
    ofxFloatSlider kangle, kx, ky; //使用顶点着色器和线段着色器

    bool showGui; //从屏幕上隐藏GUI，定义布尔变量，使用“Z“来切换

   public:
            void setup();
            void update();
            void draw();
        void stripePattern();  // 条纹图案绘制功能
        void matrixPattern();

        void draw2d();//移动所有图从draw () 到draw2d ()
```

```
49
50          void exit(); //自动保存功能
51
52              void keyPressed(int key);
53              void keyReleased(int key);
54              void mouseMoved(int x, int y );
55              void mouseDragged(int x, int y, int button);
56              void mousePressed(int x, int y, int button);
57              void mouseReleased(int x, int y, int button);
58              void mouseEntered(int x, int y);
59              void mouseExited(int x, int y);
60              void windowResized(int w, int h);
61              void dragEvent(ofDragInfo dragInfo);
62              void gotMessage(ofMessage msg);
63
64  };
```

Exercise 1: Pixel to generate image

To directly specify pixels to generate a new image or change an existing image, first create a pixel array, and then use the image.setFromPixels(data, w, h, type) method to push it into the image.

Link：https://youtu.be/5clewyYTB2I

main.cpp

```
#include "ofMain.h"
#include "ofApp.h"

//========================================================================
int main( ){

    ofSetupOpenGL(1024,768, OF_WINDOW);                    // <-----

    // this kicks off the running of my app
    // can be OF_WINDOW or OF_FULLSCREEN
    // pass in width and height too:
    ofRunApp( new ofApp());
```

```
13
14 }
```

ofApp.cpp

```
1  #include "ofApp.h"
2
3  //--------------------------------------------------------------
4  void ofApp::setup(){
5      /*
6      p = ofPoint( 100.0, 200.0 );
7      stem0 = ofPoint( 300, 100 );
8      stem1 = ofPoint( 300, 270 );
9      stem2 = ofPoint( 300, 300 );
10     stem3 = ofPoint( 300, 400 );
11     leftLeaf = ofPoint( 200, 220 );
12     rightLeaf = ofPoint( 400, 220 );
13
14     color = ofColor( 0.0, 128.0, 255.0 );
15
16     ofEnableBlendMode( OF_BLENDMODE_ADD );
17     ofEnableBlendMode( OF_BLENDMODE_ALPHA );
18
19     //image.loadImage("flower.png");
20     */
21 }
22 //--------------------------------------------------------------
23 void ofApp::update(){
24
25     //Creating image
26
27     int w = 800;  //Image width
28     int h = 800;  //Image height
29
30     //Allocate array for filling pixels data
31     unsigned char *data = new unsigned char[w * h * 4];
32
33     //Fill array for each pixel (x,y)
34     for (int y=0; y<h; y++) {
```

```
        for (int x=0; x<w; x++) {
            //Compute preliminary values,
            //needed for our pixel color calculation:

            //1. Time from application start
            float time = ofGetElapsedTimef();

            //2. Level of hyperbola value of x and y with
            //center in w/2, h/2
            float v = ( x - w/2 ) * ( y - h/2 );

            //3. Combining v with time for motion effect
            float u= v * 0.00025 + time;
            //Here 0.00025 was chosen empirically

            //4. Compute color components as periodical
            //functions of u, and stretched to [0..255]
            int red = ofMap( sin( u ), -1, 1, 0, 210 );
            int green = ofMap( sin( u * 2 ), -1, 1, 0, 180 );
            int blue = 255 - green;
            int alpha = 255;  //Just constant for simplicity

            //Fill array components for pixel (x, y):
            int index = 4 * ( x + w * y );
            data[ index ] = red;
            data[ index + 1 ] = green;
            data[ index + 2 ] = blue;
            data[ index + 3 ] = alpha;
        }
    }
     //Load array to image
       image.setFromPixels( data, w, h, OF_IMAGE_COLOR_ALPHA );
     //Array is not needed anymore, so clear memory
       delete[] data;
}

//--------------------------------------------------------------
void ofApp::draw(){
     /*
```

```
74      ofPushMatrix(); //将坐标系的原点移动到屏幕中心
75
76      ofBackground( 255, 255, 255 );              //Set white backgrou
77      //image.draw( 0, 0, 900, 600 );
78
79      //ofSetColor( 0, 255, 0 );
80      ofSetColor( 255, 255, 255, 255 );
81      //image.draw( 150, 0, 900, 600 );
82
83      image.draw( 10, 0 );
84      image.draw( 20, 0 );
85
86      ofSetColor( 255, 255, 255, 128 );
87      image.draw( 40, 0 );
88      */
89      ofBackground(255, 255, 255);     //Set up white background
90      ofSetColor( 255, 255, 255 );     //Set color for image drawing
91      image.draw( 0, 0 );              //Draw image
92      /*
93      ofSetColor( 0, 0, 0 );                     //Set black color
94      ofCircle( stem0, 40 );                  //Blossom
95      ofLine( stem0, stem3 );                 //Stem
96      ofTriangle( stem1, stem2, leftLeaf );   //Left leaf
97      ofTriangle( stem1, stem2, rightLeaf );  //Right leaf
98      */
99      //ofPopMatrix();
100 }
```

ofApp.h

```
1  #pragma once
2  #include "ofMain.h"
3
4  class ofApp : public ofBaseApp{
5      ofPoint p;
6      ofPoint stem0, stem1, stem2, stem3, leftLeaf, rightLeaf;
7      ofColor color;
8      ofImage image;
9     public:
```

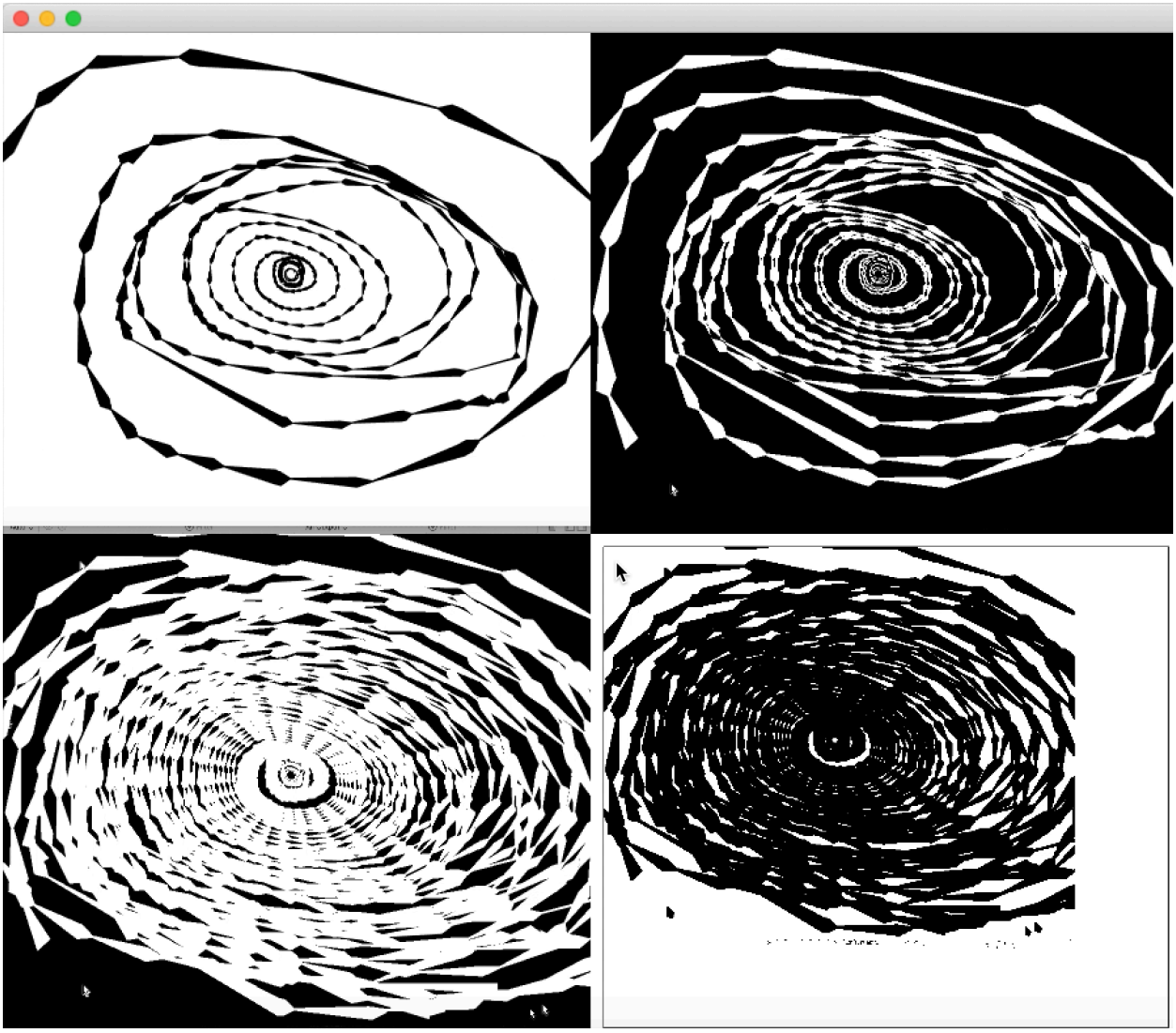
```
10              void setup();
11              void update();
12              void draw();
13
14              void keyPressed(int key);
15              void keyReleased(int key);
16              void mouseMoved(int x, int y);
17              void mouseDragged(int x, int y, int button);
18              void mousePressed(int x, int y, int button);
19              void mouseReleased(int x, int y, int button);
20              void mouseEntered(int x, int y);
21              void mouseExited(int x, int y);
22              void windowResized(int w, int h);
23              void dragEvent(ofDragInfo dragInfo);
24              void gotMessage(ofMessage msg);
25  };
```

Creative motivation:

I saw how the dynamic color waves of raster images work. It can use the image.getPixels() function to change the color image file. When creating the image, I can try more effects from this direction.

Exercise 2: Use ofxC to process image by image

The image is initialized first, and then the image pixels are manipulated through algebraic operations, and then the image is drawn on the screen through the draw function, which uses the function of color conversion between plane and space. After considering the motion detector, the current movie frame in grayscale, absolute difference of the current and the previous frame, the accumulated buffer, and finally, the motion areas shown in black color are finally displayed.



Link：https://youtu.be/m9TECAumTuw

main.cpp

```
#include "ofMain.h"
#include "ofApp.h"

//========================================================================
int main( ){
    ofSetupOpenGL(1024,768,OF_WINDOW);                    // <-----

    // this kicks off the running of my app
    // can be OF_WINDOW or OF_FULLSCREEN
    // pass in width and height too:
    ofRunApp(new ofApp());

}
```

ofApp.cpp

```cpp
#include "ofApp.h"

//--------------------------------------------------------------
void ofApp::setup(){
    video.loadMovie( "try.mov" );  //Load the video file
    video.play();                       //Start the video to play
}

//--------------------------------------------------------------
void ofApp::update(){
    video.update();  //Decode the new frame if needed
    //Do computing only if the new frame was obtained
    if ( video.isFrameNew() ) {
        //Store the previous frame, if it exists till now
        if ( grayImage.bAllocated ) {
            grayImagePrev = grayImage;
        }

        //Getting a new frame
        image.setFromPixels( video.getPixelsRef() );
        grayImage = image;  //Convert to grayscale image

        //Do processing if grayImagePrev is inited
        if ( grayImagePrev.bAllocated ) {
            //Get absolute difference
            diff.absDiff( grayImage, grayImagePrev );

            //We want to amplify the difference to obtain
            //better visibility of motion
            //We do it by multiplication. But to do it, we
            //need to convert diff to float image first
            diffFloat = diff;   //Convert to float image
            diffFloat *= 5.0;   //Amplify the pixel values

            //Update the accumulation buffer
            if ( !bufferFloat.bAllocated ) {
                //If the buffer is not initialized, then
```

```
                //just set it equal to diffFloat
                bufferFloat = diffFloat;
            }
            else {
                //Slow damping the buffer to zero
                bufferFloat *= 0.8;
                //Add current difference image to the buffer
                bufferFloat += diffFloat;
            }
        }
    }

}

//--------------------------------------------------------------
void ofApp::draw(){
    ofBackground( 255, 255, 255 );  //Set the background color

    //Draw only if diffFloat image is ready.
    //It happens when the second frame from the video is obtained
    if ( diffFloat.bAllocated ) {
        //Get image dimensions
        int w = grayImage.width;
        int h = grayImage.height;

        //Set color for images drawing
        ofSetColor( 255, 255, 255 );

        //Draw images grayImage,  diffFloat, bufferFloat
        grayImage.draw( 0, 0, w/4, h/4 );
        diffFloat.draw( w/4 , 0, w/4, h/4 );
        bufferFloat.draw( 0, h/4, w/4, h/4 );

        //Draw the image motion areas
        //Shift and scale the coordinate system
        ofPushMatrix();
        ofTranslate( w/4+10, h/4+10 );
        ofScale( 0.2, 0.2 );

```

```
        //Draw bounding rectangle
        ofSetColor(0, 0, 0);
        ofNoFill();
        ofRect( 0, 0, w*1.2, h*1.2);

        //Get bufferFloat pixels
        float *pixels = bufferFloat.getPixelsAsFloats();
        //Scan all pixels
        for (int y=0; y<h; y++) {
            for (int x=0; x<w; x++) {
                //Get the pixel value
                float value = pixels[ x + w * y ];
                //If value exceed threshold, then draw pixel
                if ( value >= 0.1) {
                    ofRect( x, y, 1, 1 );
                    //Rectangle with size 1x1 means pixel
                    //Note, this is slow function,
                    //we use it here just for simplicity
                }
            }
        }
        ofPopMatrix();    //Restore the coordinate system
    }
}
```

ofApp.h

```
#pragma once

#include "ofMain.h"
#include "ofxOpenCv.h"

class ofApp : public ofBaseApp{

    ofVideoPlayer video;         //Declare the video player object
    ofxCvColorImage image;          //The current video frame
    //The current and the previous video frames as grayscale images
    ofxCvGrayscaleImage grayImage, grayImagePrev;
    ofxCvGrayscaleImage diff;    //Absolute difference of the frames
```

```
    ofxCvFloatImage diffFloat;   //Amplified difference images
    ofxCvFloatImage bufferFloat; //Buffer image

   public:
            void setup();
            void update();
            void draw();

            void keyPressed(int key);
            void keyReleased(int key);
            void mouseMoved(int x, int y );
            void mouseDragged(int x, int y, int button);
            void mousePressed(int x, int y, int button);
            void mouseReleased(int x, int y, int button);
            void mouseEntered(int x, int y);
            void mouseExited(int x, int y);
            void windowResized(int w, int h);
            void dragEvent(ofDragInfo dragInfo);
            void gotMessage(ofMessage msg);
};
```

Creative motivation:

Compared with the previous exercise, the advantage of using openCV for image pixel processing is that when performing more complex image processing, in pixel-by-pixel programming, using openCV, most image operations can be completed with only one line of code, and Can improve the performance of the project.

4. Evaluation and summary

A. Some laws learned

The overall image processing function:

1. image.resize(newW, newH) – resizes the image to a new size, newW × newH
2. image.crop(x, y, w, h) – crops the image to a subimage with the top-left corner

(x, y) and size w × h
3. image.rotate90(times) – rotates the image clockwise at 90 * times degrees
4. image.mirror(vertical, horizontal) – mirrors the image,
where vertical and horizontal are bool values
5. image2.clone(image) – copies image into image2

Three classes for working with images :
1. ofImage: This is intended for manipulating the pixel values and drawing an image
2. ofPixels: This is intended for manipulating the pixels values only
3. ofTexture: This is intended for drawing images only

B. Among the technical points to achieve the goal, I am more satisfied with:
With the help of learning materials, I implemented a video mixer and learned the basics of raster graphics, about how to load and draw raster images and videos, and how to capture video from the camera.

C. Locations that can be enhanced and improved
At present, the implementation effect of the project is still not under my control. I still have a lot of content to learn in the study of basic C++ code and framework. I hope to use ofGUI plug-in and openCV plug-in to explore more effects of computational art effects.

5. The next step of the project
I found that there are many infrequent plug-ins on the official website of openframework, and I want to download an ARKit plug-in and make an AR app.