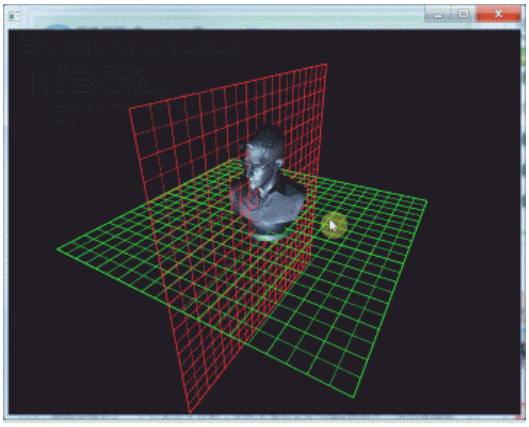
ofxManipulatorとMSAInterpolatorを組み合わせて、3次元スプラインをお洒落に操作するインターフェースのExample

おとといぐらいから紹介している記事を組み合わせて、ofxManipulatorとMSAInterpolatorがいかに便利かを示すExampleを作ります。



プロジェクトに必要なaddonは、
ofxAssimpModelLoader,
ofxManipulator(改造版)
ofxMSAInterpolator(改造版)
です。
2つのExampleをくっつけただけですが、とても簡単に良い感じの3次元インターフェースが作れます。
この例では，3次元スプラインの制御点をマウスでぐりぐり視点を切り替えながら編集できる機能を作っています。
操作は、

- マウスの右ボタンで、編集かカメラ視点移動かを切り替え。
- マウスの左ボタンダブルクリックで、スプラインの制御点を追加。
- マウスのスクロールで、編集の移動、回転、拡大縮小の切り替え。ただし、この例では移動のみ使用。
- キーボード1~5のキーで補間の計算方法を変更

main.cpp

```
#include "ofApp.h"

int main()
{
  ofGLFWWindowSettings settings;
  settings.setGLVersion(2,1);
  auto window = ofCreateWindow(settings);
  auto app = make_shared<ofApp>();
  ofRunApp(window, app);

  return ofRunMainLoop();
}
```

ofApp.h

cvl-robot

読者になる 62

このブログについて

検索

記事を検索

リンク

はてなブログ

ブログをはじめる

週刊はてなブログ

はてなブログPro

最新記事

メモ："qt.qpa.plugin: C
find the Qt platform plug
"windows" in "" error" が
Windows環境vcpkgでop
インストールしたときに
てしまった場合の対処法

オムロンの環境センサ2
BU01の公式サンプルに
と足して、シリアルナン
取得できるようにする

掘り起こし手直し版　c
とofxTurboJpegで画像配
信

買う予定の本2022年7月

Arduino MKR IoT Carrie
Grove SCD30搭載CO2
使って、長期間の環境測
てみる。

月別アーカイブ

▶ 2022 (4)
▶ 2021 (7)
▶ 2020 (7)
▶ 2019 (15)
▶ 2018 (72)
▶ 2017 (31)
▶ 2016 (46)

```
#pragma once

#include "ofMain.h"

#include "ofxAssimpModelLoader.h"
#include "ofxManipulator.h"
#include "MSAInterpolator.h"

class ofApp : public ofBaseApp
{
  public:
    void setup();
    void draw();
    void keyPressed(ofKeyEventArgs&);
    void mousePressed(ofMouseEventArgs&);

    ofLight m_light;
    ofImage m_matcapImage;
    ofShader m_matcapShader;
    ofMatrix4x4 m_modelMatrix;
    ofxAssimpModelLoader m_model;
        ofEasyCam m_cam;
        //ofxManipulator m_gizmo;

        void mouseReleased(ofMouseEventArgs & mouse);

        // double clicke
        unsigned long lastTap;
        void mouseDoubleClicked(ofMouseEventArgs &mouse);
        ofEvent<ofMouseEventArgs&> doubleClickEvent;

        // ofxManipulator
        vector<shared_ptr<ofxManipulator> > gizmo;
        typedef std::shared_ptr<ofxManipulator> M_Ptr;
        M_Ptr createManipulator() {     return M_Ptr(new ofxManipulator()); };
        void onTranslateChange(ofxManipulatorEventArgs& m);

        // MSAInterpolator
        msa::Interpolator2D     spline2D;
        msa::Interpolator3D     spline3D;

        msa::InterpolationType  interpolationType;
        bool                    useLength;

        float                   currentRot;
        bool                    rotateView;

        float                   spherePosPerc;                  // 0....1 percentage of how far along the 3D
        float                   sphereSpeed;
};
```

ofApp.cpp

```
#include "ofApp.h"

void ofApp::setup()
{
  ofSetWindowPosition(
    ( ofGetScreenWidth() -  ofGetWidth()) / 2.0f,
    (ofGetScreenHeight() - ofGetHeight()) / 2.0f
  );

  ofDisableArbTex();
  ofEnableLighting();
  ofSetBackgroundColor(28, 28, 38);

 // m_cam.movespeed = 3.0f;
  m_cam.setNearClip(5.0f);
```


▼ 2015 (43)
- 2015 / 12 (9)
- 2015 / 9 (1)
- 2015 / 8 (13)
- 2015 / 7 (5)
- 2015 / 6 (4)
- 2015 / 4 (2)
- 2015 / 3 (4)
- 2015 / 2 (3)
- 2015 / 1 (2)

▶ 2014 (68)
▶ 2013 (45)

参加グループ

テクノロジー

その他

```
  m_cam.setFarClip(10000);
  m_cam.move(120, 120, 250);
  m_cam.lookAt(ofVec3f(0, 120, 0));
  m_cam.setDistance(25.0f);

  m_light.setDirectional();
  m_light.rotate(180, 0, 1, 0);

  m_modelMatrix.scale(1, 1, 1);
  m_modelMatrix.rotate( 90, 1, 0, 0);
  m_modelMatrix.rotate(165, 0, 1, 0);
  m_modelMatrix.translate(0, 140/30, 0);

  m_matcapImage.load("MatCap.png");
  m_matcapShader.load("MatCap.vs", "MatCap.fs");

  m_model.loadModel("Edward_Joseph_Snowden.obj");

  // double click
  lastTap = 1000;
  ofAddListener(doubleClickEvent, this, &ofApp::mouseDoubleClicked);

  // MSAInterpolator
  interpolationType = msa::kInterpolationCubic;
  useLength = false;
  spherePosPerc = 0;
  sphereSpeed = 0.005f;
}

void ofApp::draw()
{
  m_cam.begin();

    ofDisableLighting();
      ofDrawGrid(50/30, 10, false, true, true, false);
    ofEnableLighting();

        // MSAInterpolator
        int numSteps = floor(mouseX / (float)ofGetWidth() * 1000);
        if (numSteps<10) numSteps = 10;

        ofDisableLighting();
        // draw raw spline3D
        glColor3f(1, 1, 1);
        drawInterpolatorRaw(spline3D);

        // draw interpolated spline3D
        ofColor c = ofColor::yellow;
        glColor3f(c.r, c.g, c.b);
        drawInterpolatorSmooth(spline3D, numSteps);

        // draw sphere moving along 3D path
        ofVec3f spherePos = spline3D.sampleAt(spherePosPerc);
        glPushMatrix();
        glColor3f(1, 1, 0);
        glTranslatef(spherePos.x, spherePos.y, spherePos.z);

        m_light.enable();

        m_matcapShader.begin();
        m_matcapShader.setUniformTexture("litsphereTexture", m_matcapImage, 1);
        ofPushMatrix();
        ofMultMatrix(m_modelMatrix); // *m_gizmo.getMatrix());
        m_model.getMesh("").drawFaces();
        ofPopMatrix();
        m_matcapShader.end();

        m_light.disable();

        glPopMatrix();
```

```
        // move sphere
        // if it reaches the edges, bounce back
        spherePosPerc += sphereSpeed;
        if (spherePosPerc > 1) {
                spherePosPerc = 1;
                sphereSpeed *= -1;
        }
        else if (spherePosPerc < 0) {
                spherePosPerc = 0;
                sphereSpeed *= -1;
        }

        // ofxManipulator
        ofDisableDepthTest();
        ofDisableLighting();
        for (int i = 0; i < gizmo.size(); i++) {
                shared_ptr<ofxManipulator> m_ptr = gizmo[i];
                m_ptr->draw(m_cam);
        }
        ofEnableLighting();

  m_cam.end();

  ofPushStyle();
  ofDisableDepthTest();
  ofSetColor(ofColor::white);
  string uiLin = interpolationType == msa::kInterpolationLinear ? "* " : "  ";
  string uiCub = interpolationType == msa::kInterpolationCubic ? "* " : "  ";
  string uiCos = interpolationType == msa::kInterpolationCosine ? "* " : "  ";
  string uiCat = interpolationType == msa::kInterpolationCatmullRom ? "* " : "  ";
  string uiHer = interpolationType == msa::kInterpolationHermite ? "* " : "  ";
  string uiDist = spline3D.getUseLength() ? "* " : "  ";
  ofDrawBitmapString(ofToString(ofGetFrameRate(), 2) + "\n"
          + "numSteps (resampling resolution - mouseX to change): " + ofToString(numSteps) + "\n"
          + "mouse click around the area to draw a 3D spline (length = " + ofToString(spline3D.getLength())
          + "\n"
          + uiLin + "'1' to use linear interpolation\n"
          + uiCub + "'2' to use cubic (catmull rom) interpolation\n"
          + uiCos + "'3' to use cosine interpolation\n"
          + uiCat + "'4' to use catmull rom interpolation\n"
          + uiHer + "'5' to use hermite interpolation with random tension and bias\n"
          + "\n"
          + uiDist + "'d' to toggle 'using Length in interpolation'\n"
          + "\n"
          + "'c' to clear 3D spline\n"
          , 20, 20);
  ofEnableDepthTest();
  ofPopStyle();
}

void ofApp::keyPressed(ofKeyEventArgs &key)
{
  switch (key.keycode) {
          case '1':
                  interpolationType = msa::kInterpolationLinear;
                  spline3D.setInterpolation(interpolationType);
                  spline2D.setInterpolation(interpolationType);
                  break;
          case '2':
                  interpolationType = msa::kInterpolationCubic;
                  spline3D.setInterpolation(interpolationType);
                  spline2D.setInterpolation(interpolationType);
                  break;
          case '3':
                  interpolationType = msa::kInterpolationCosine;
                  spline3D.setInterpolation(interpolationType);
                  spline2D.setInterpolation(interpolationType);
                  break;
          case '4':
```

```
                interpolationType = msa::kInterpolationCatmullRom;
                spline3D.setInterpolation(interpolationType);
                spline2D.setInterpolation(interpolationType);
                break;
        case '5':
                interpolationType = msa::kInterpolationHermite;
                spline3D.setInterpolation(interpolationType);
                spline2D.setInterpolation(interpolationType);
                spline3D.setTension(ofRandom(-1, 1));
                spline3D.setBias(ofRandom(-1, 1));
                spline2D.setTension(spline3D.getTension());
                spline2D.setTension(spline3D.getBias());
                cout << "tension = " << spline3D.getTension() << ", bias = " << spline3D.getBias() << endl
                break;
        case 'd':
                useLength ^= true;
                spline3D.setUseLength(useLength);
                spline2D.setUseLength(useLength);
                break;

        case 'c':
        case 'C':
                spline3D.clear();
                break;

        case 'r':
        case 'R':
                rotateView ^= true;
                break;
        }
}

void ofApp::mousePressed(ofMouseEventArgs &mouse)
{
  switch (mouse.button)
  {
        case (OF_MOUSE_BUTTON_RIGHT) : {
                if (m_cam.getMouseInputEnabled()) m_cam.disableMouseInput();
                else m_cam.enableMouseInput();
        }
  }
}

void ofApp::mouseReleased(ofMouseEventArgs & mouse) {
        // for double clicing
        static const unsigned long doubleclickTime = 400;
        unsigned long curTap = ofGetElapsedTimeMillis();
        if (lastTap != 0 && curTap - lastTap < doubleclickTime) {
                ofNotifyEvent(doubleClickEvent, mouse);
                return;
        }
        lastTap = curTap;
}

void ofApp::mouseDoubleClicked(ofMouseEventArgs & mouse) {
        if (mouse.button == OF_MOUSE_BUTTON_1) {
                // MSAInterpolator
                // when you add a new point, the length of the spline increases
                // so the sphere will jump to a new position because it's position is based on percentage
                // so a little bit of maths to calculate what the new position percentage should be to stay
                // (not precise, but close enough)
                int numPoints = spline3D.size();
                spherePosPerc = spherePosPerc * numPoints / (numPoints + 1);

                ofVec3f pt(mouse.x, mouse.y, 0);
                ofVec3f wrpt = m_cam.screenToWorld(pt);
                spline3D.push_back(wrpt);

                // ofxManipulator
                gizmo.push_back(createManipulator());
```

```
                int id = spline3D.size() - 1;
                shared_ptr<ofxManipulator> m_ptr = gizmo.back();
                m_ptr->setID(spline3D.size() - 1);
                m_ptr->setTranslation(wrpt);
                m_ptr->setAutoFocus(true);

                ofAddListener(m_ptr->onTranslateChange, this, &ofApp::onTranslateChange);
        }
}

void ofApp::onTranslateChange(ofxManipulatorEventArgs& m)
{
        ofVec3f& vec = spline3D.at(m.id);
        vec = m.translation;
}
```

TODO

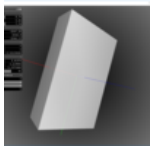
- グループ選択の有効化: CTRLキーを押しながら選択したターゲットはまとめて操作できるようにしたい。
- 排他的選択の有効化:　一番近いターゲットだけにフォーカスを合わせる選択も選べるようにしたい。
- スケールの補間
- 姿勢（回転）の補間
- Splineの制御点の端点だけではなく途中への挿入、消去の編集
- Splineのファイルへの入出力

今日の漫画

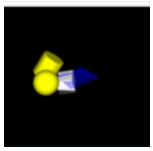
最近面白いご飯漫画がとても多いですね。その中でもこのどんぶり委員長は、食欲に訴えかける力の強いマンガです。

cvl-robot 6年前

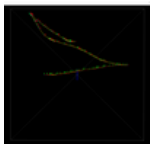
関連記事



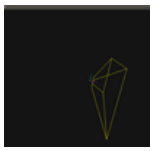
2018-08-06
安価なIMU(AHRS)センサのBosch BNO055USBStickを試す（その３）
前回、BNO055USBStickのデータを自前のプログラムで取得するこ…



2018-05-21
動力学シミュレータofxBulletを静的な衝突判定ライブラリとしてだけ使う方法
OMPLが自由に使えるようになって経路計画がお手軽で楽しいので…



2018-05-10
openFrameworksでOMPLを動かしたい（その4）
omplのdemo_RigidBodyPlanningWithControlsをopenFrameworksに…



2018-01-24
openFrameworksをROSのフロントエンドにする(その3)
出来上がったのがこちら。データさえ拾えれば、ROS関係なくて普…

2017-05-15
ofFboにofTextureから大きな画像の一部を斜めに切り出す方法
大きな画像から、任意の位置と角度で、適当な矩形を切り出す方…

コメントを書く

« Texの編集にはCloudLatexが便利

openFrameworksでマウスボタンのdoubleCli… »