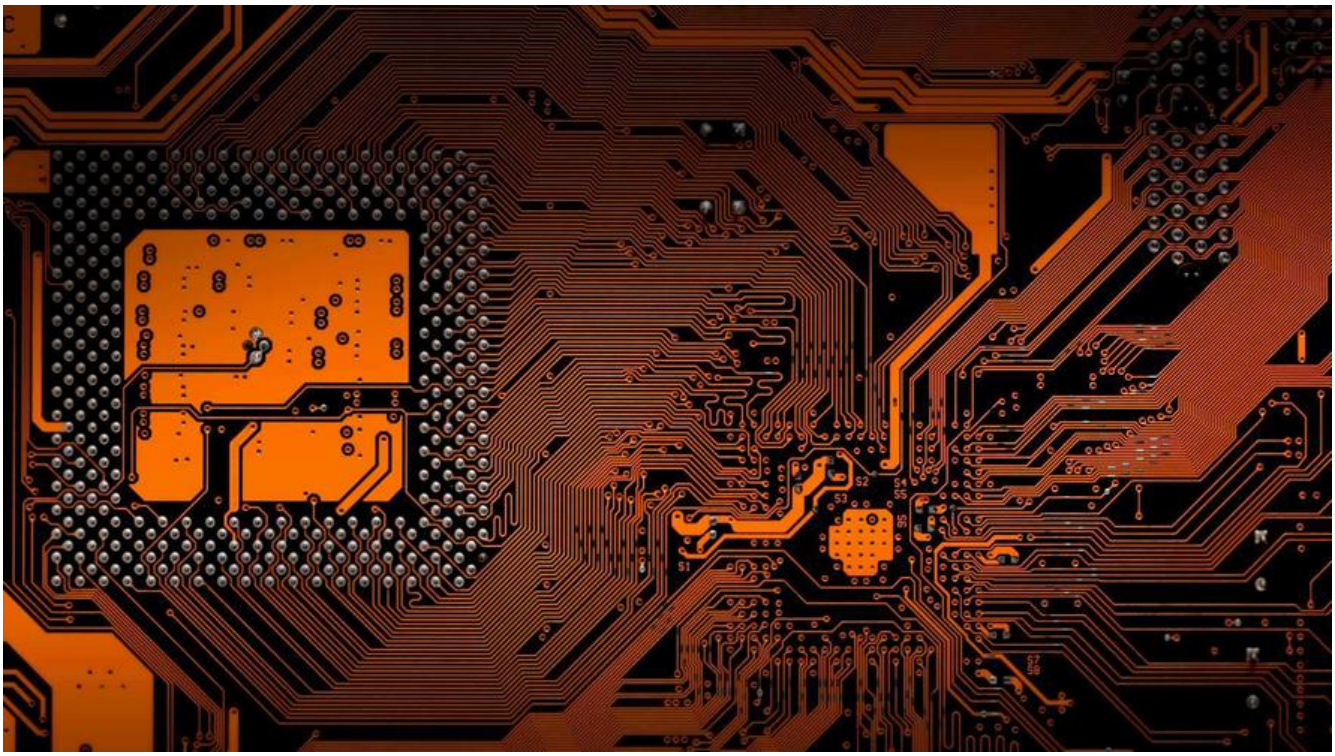
链滴

VEX-IQ 3D 模型解决方案

作者：yang17762622

原文链接：https://ld246.com/article/1569314269490

来源网站：链滴



一、前言

近期有个小程序+pc的项目，需要将客户的VEX-IQ模型做成3D效果
经过一番研究，最终采用Threejs实现成功，记录一下...

二、准备工具

2.1 SnapCAD

客户编辑VEX-IQ模型所用工具，生成的文件为.ldr
主要用来安装后获取里面的零件库
下载地址：SnapCAD

2.2 Blender 2.80

Blender用于将.ldr转换成Three目前主流支持的格式为gftl/glb
下载地址：Blender

2.3 ImportLDRaw

Blender默认不支持.ldr导入，ImportLDRaw为Blender识别.ldr文件插件。
此插件包含两个部分，一个为导入插件，使Blender支持LDRaw导入。
另一个为零件库，主要负责导入进来的模型颜色和图案显示
操作说明：https://github.com/TobyLobster/ImportLDraw
插件zip：https://github.com/TobyLobster/ImportLDraw/releases

零件库：https://ldraw.org/parts/latest-parts.html

2.4 ThreeJs

一个非常好用的实现浏览器端3D效果的工具，用于渲染Blender导出后的模型文件

下载地址：https://threejs.org/

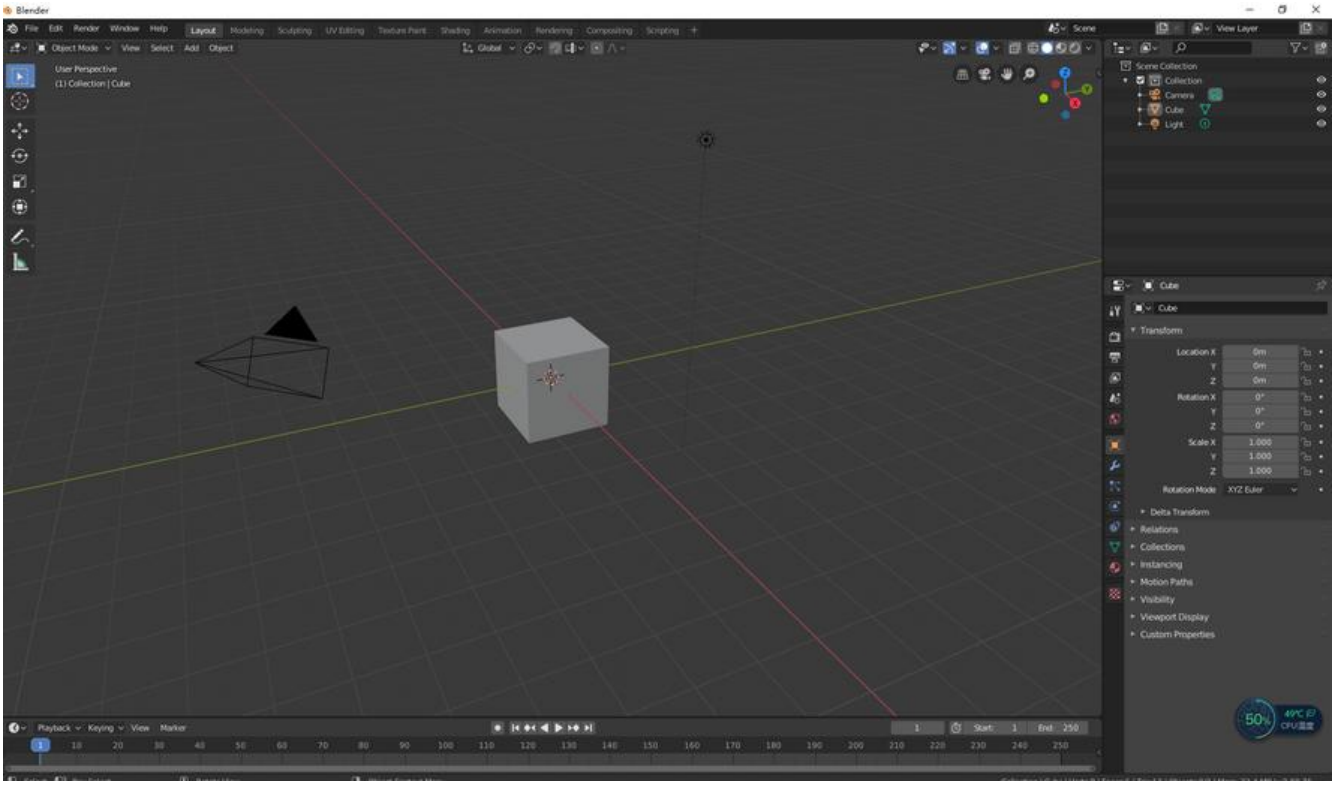
三、安装工具和插件

3.1 安装SnapCAD 和Blender

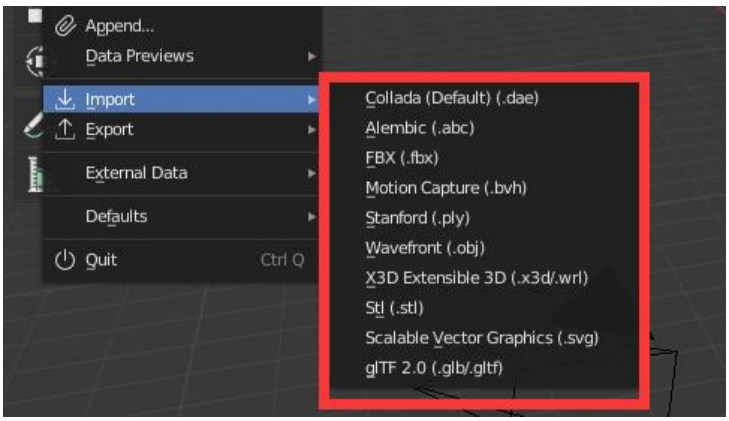
将下载好的安装文件安装，直接下一步到成功，非常简单。

3.2 安装ImportLDRaw插件

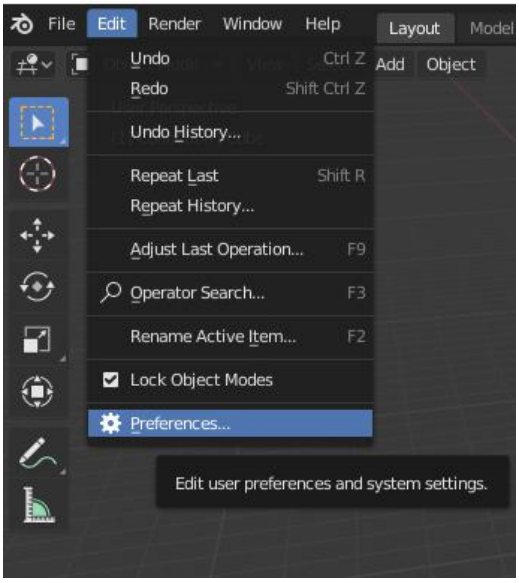
打开Blender



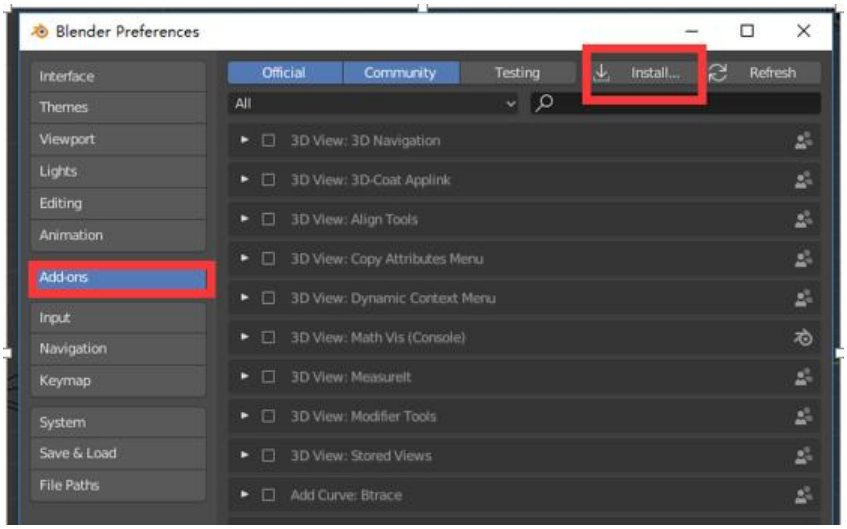
默认是不支持导入.ldr格式文件的



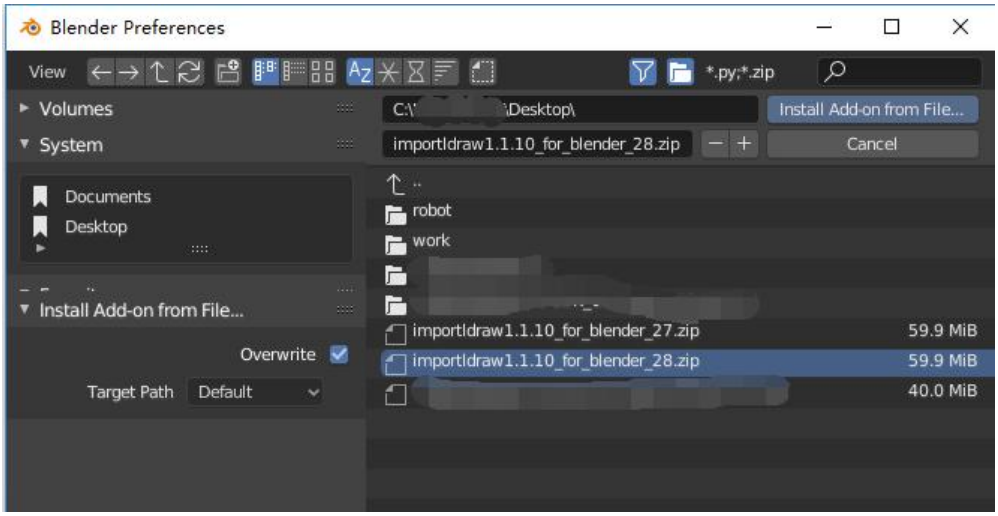
选择菜单Edit – Preferences 进入设置画面



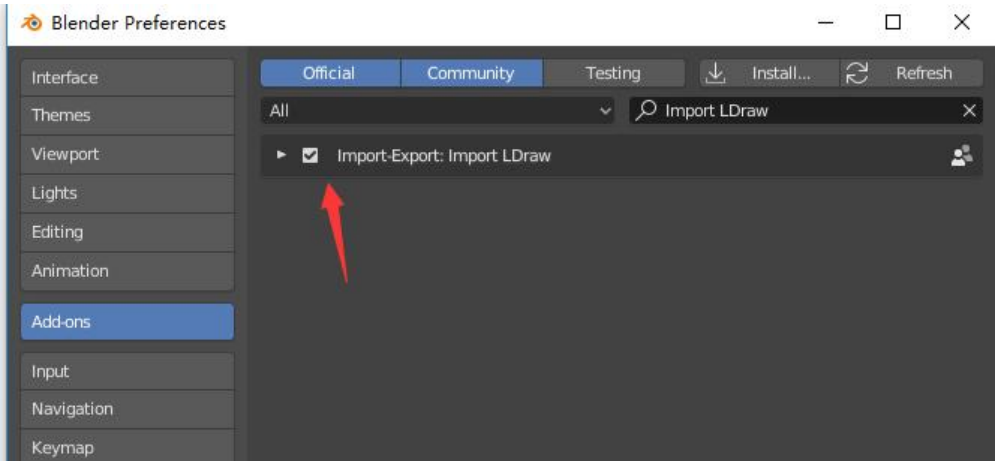
选择Add-on -- Install.. 进入添加插件设置画面



选择下载好的ImportLDRaw插件ZIP包，双击或者点击Install Add-on ... 按钮

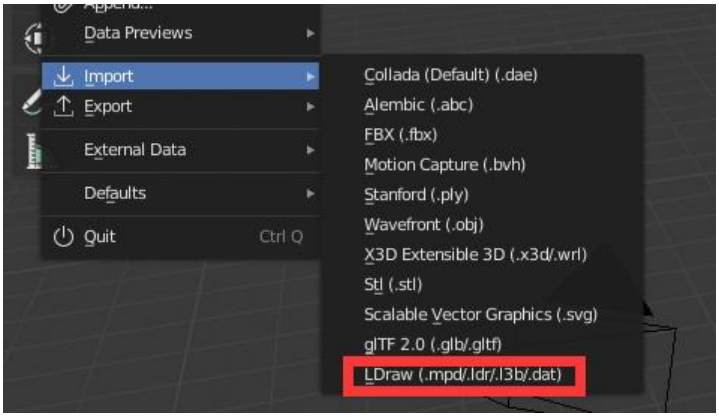


勾选 ImportLDRaw，勾选后才会生效



插件验证，选择File -- Import

已经支持导入.ldr文件

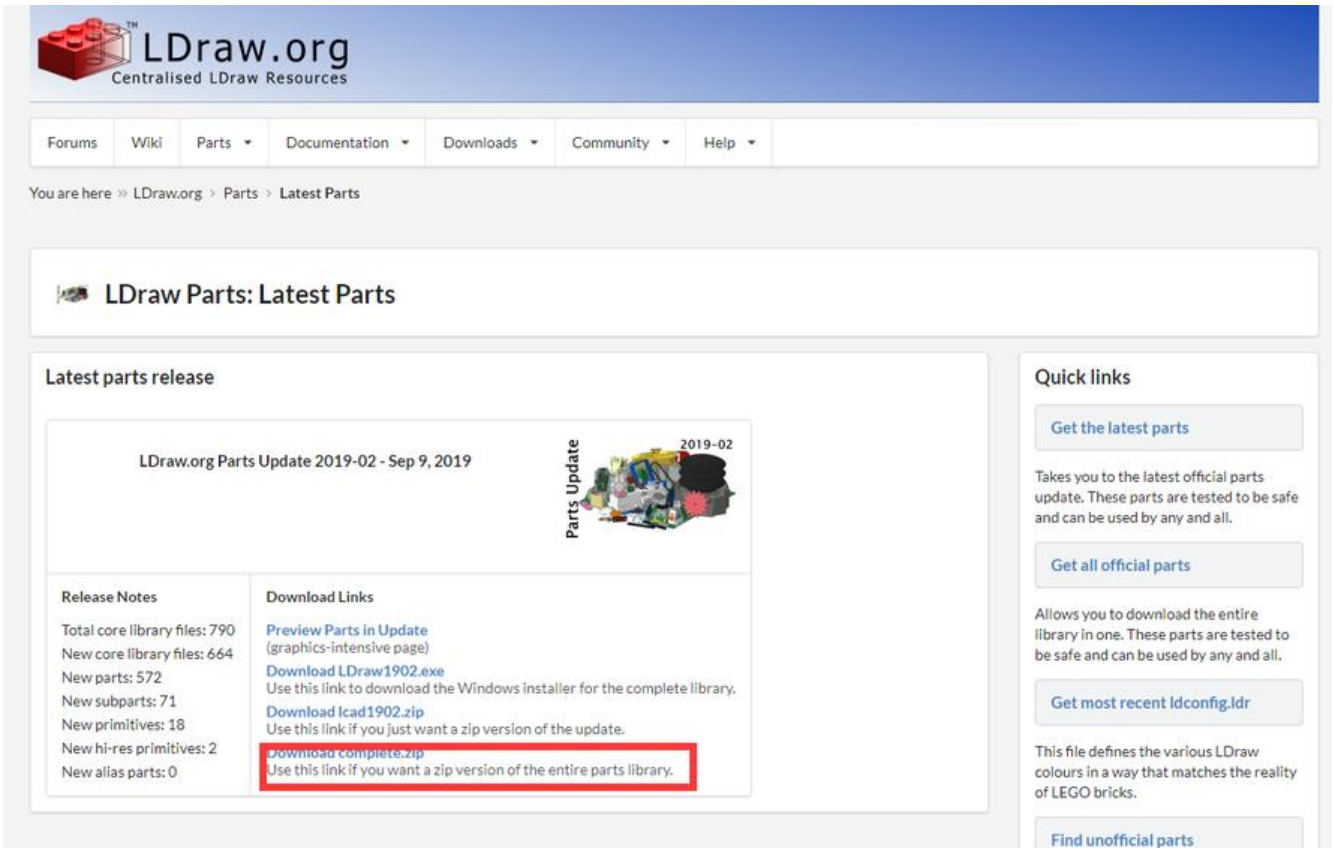


3.2 设置零件库

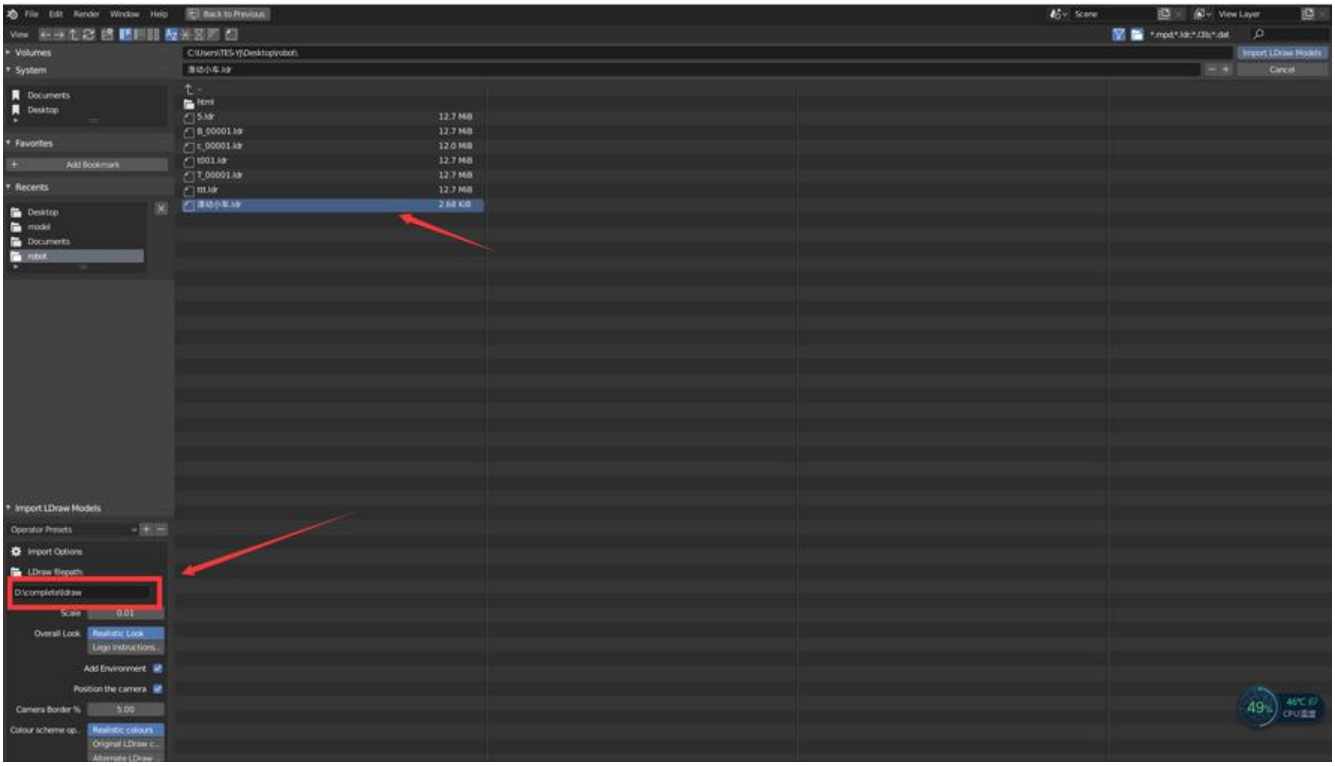
若不设置该步骤，.ldr无法在Blender中正常显示

前往https://ldraw.org/parts/latest-parts.html下载完整零件库包，下载后随便解压到一个目录下。

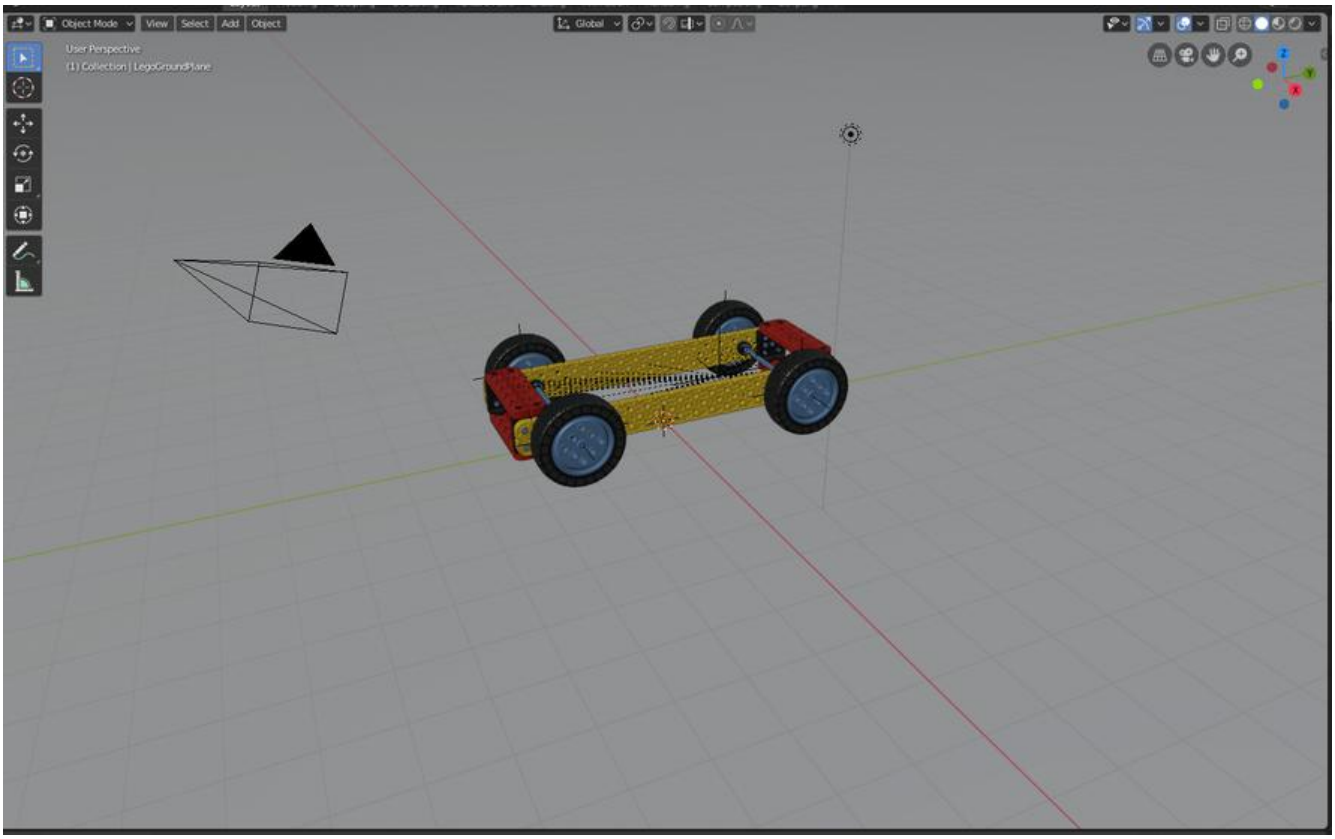
解压完成后，将SnapCAD下的p和parts文件夹复制到零件库ldraw文件夹下，重复文件选择跳过，该作用于.ldr颜色渲染，不执行会导致.ldr在Blender中没有颜色



在Blender中选择导入，类型选择LDRaw，在左下角Import Options中LDRaw path 填入刚刚解压的件库地址



导入成功后效果如下



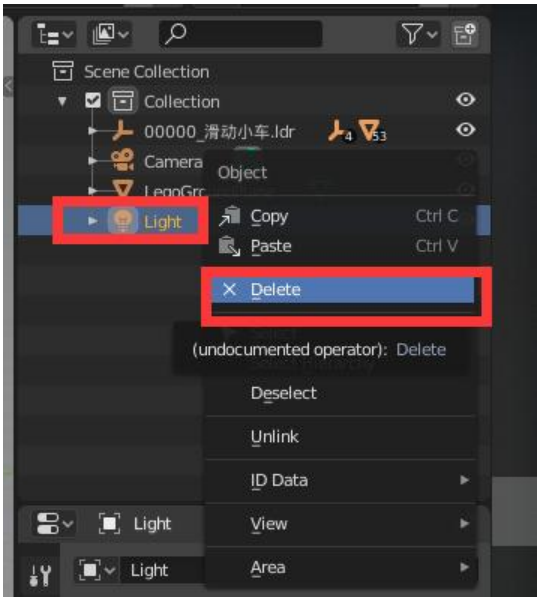
四、导出模型且渲染（FBX）

4.1 导出模型

删除灯光Light和地平面LegoGroundPlane（建议）

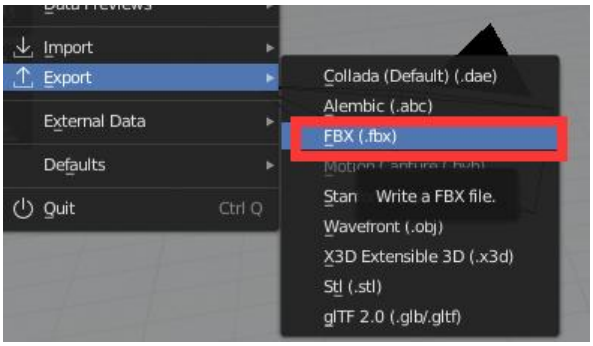
因为如果灯光位置不好，会导致整个模型太亮而发白，而导入时模型可能会在地平面之下，需要将模手动移到地平面上面，否则导出的模型会是空白的。所以建议删除这两项（个人操作下来的结果，也能是我不会用）

右上角选择Light 右键菜单选择Delete删除灯光，LegoGroundPlane也是同样操作



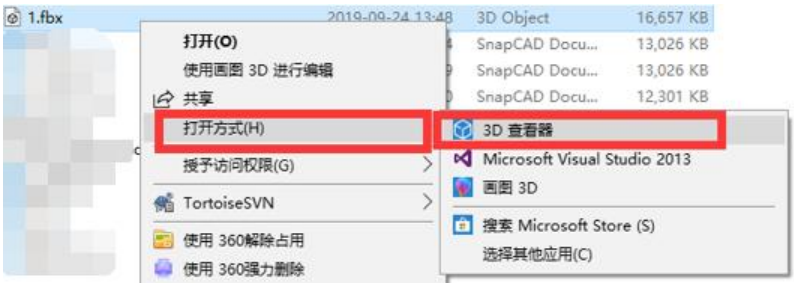
导出fbx，选择File – Export – FBX

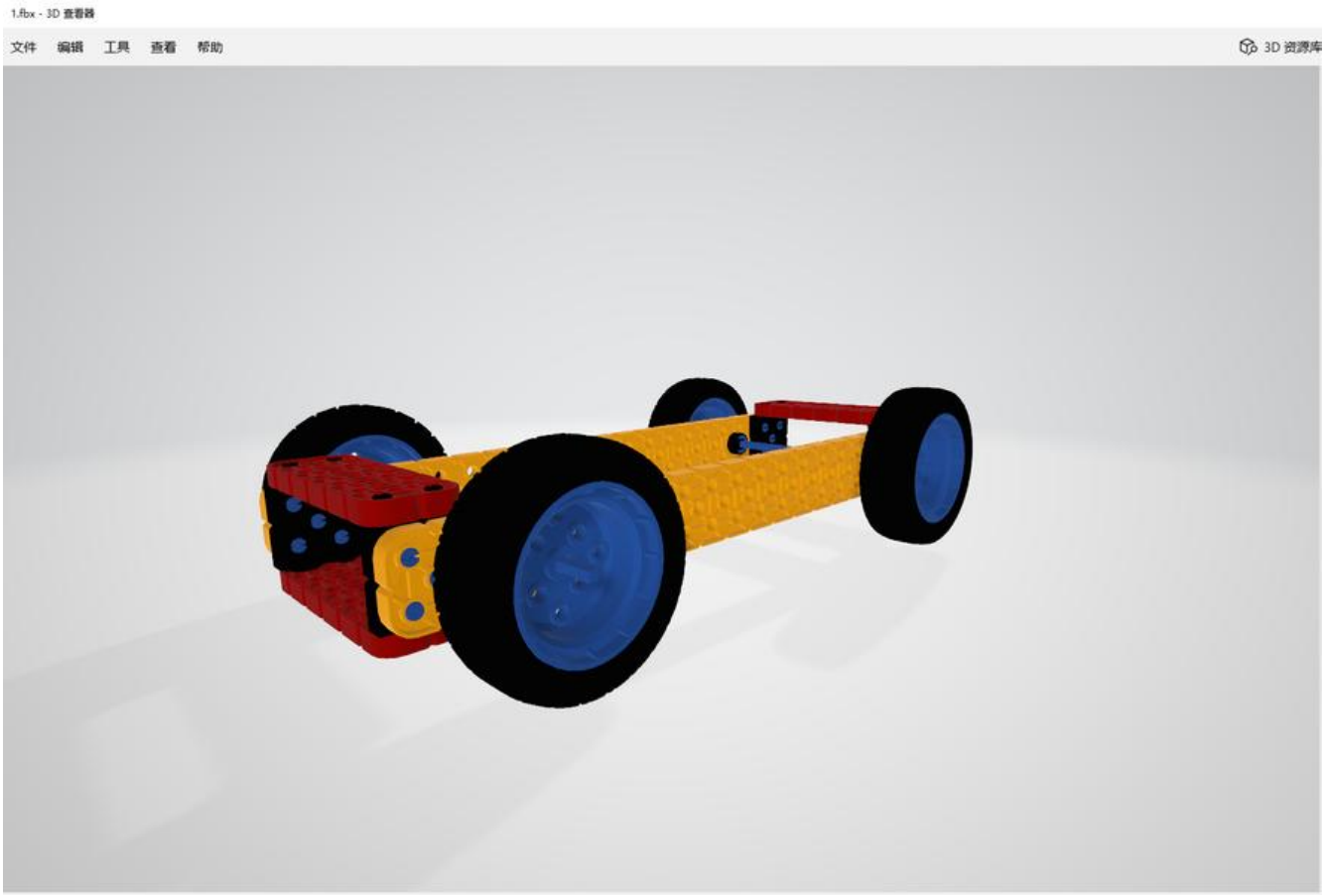
threeJS主要建议使用gftl或者glb格式，但是本次项目.ldr文件即使所有零件库都已加载，导出gftl或glb还是会纯白色。而fbx则是正常显示的。



选择好保存路径和文件名，点击Export FBX按钮，等待Blender返回到模型展示页面即可

查看导出结果，若是win10，可以用自带的3D查看器查看效果

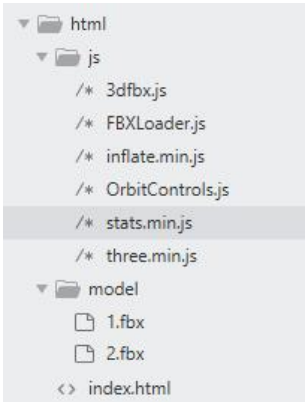




五、渲染模型

参照ThreeJS官方文档https://threejs.org/docs/index.html#manual/en/introduction/Creating-a-cene

实例代码结构



index.html

```
<html lang="en">
  <head>
    <title>FBX-3D模型观看</title>
    <meta charset="utf-8">
  </head>
  <body>
    <script type="application/javascript">
```

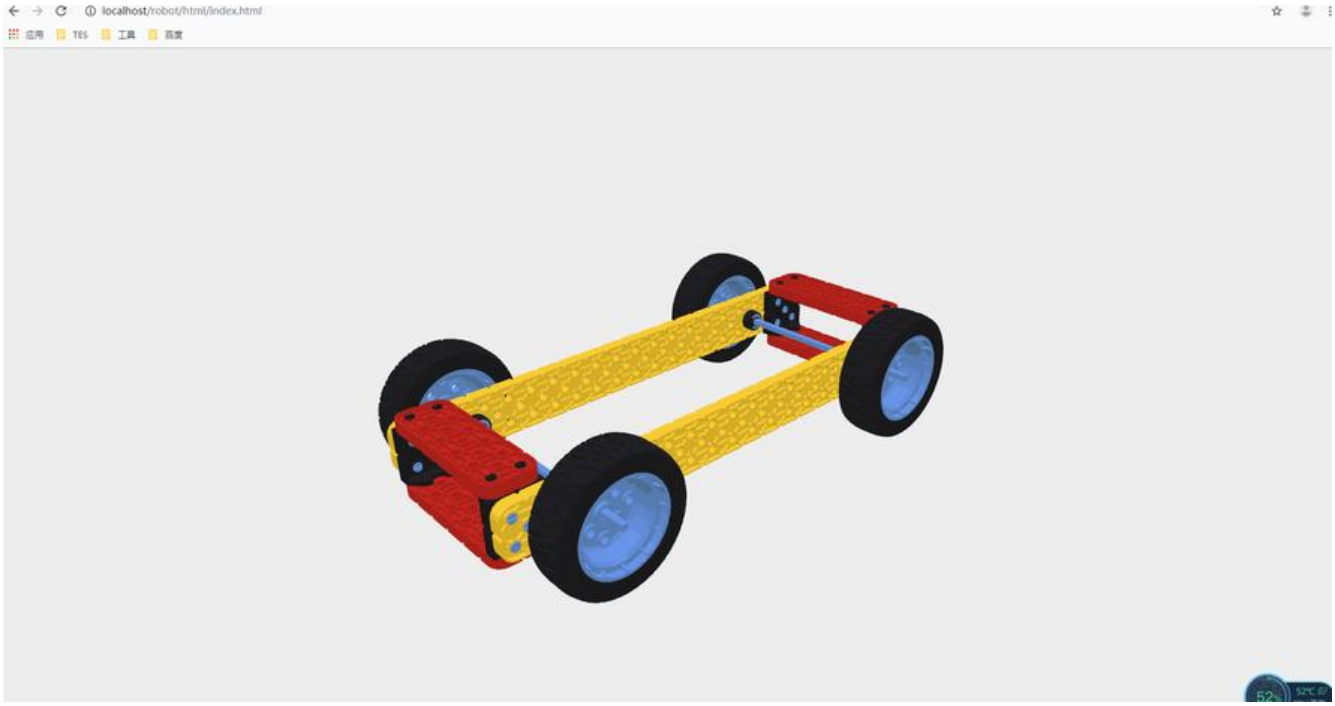
```
        //定义模型路径
        var fbxUrl = 'model/1.fbx'
    </script>
    <script src="js/three.min.js?v=1.0.0"></script>
    <script src="js/inflate.min.js?v=1.0.0"></script>
    <script src="js/OrbitControls.js?v=1.0.0"></script>
    <script src="js/FBXLoader.js?v=1.0.0"></script>
    <script src="js/stats.min.js?v=1.0.0"></script>
    <script src="js/3dfbx.js?v=1.0.0"></script>
  </body>
</html>
```

3dfbx.js

```
//3dfbx.js

var container, stats, controls;
var camera, scene, renderer, light, bbox;
var rotating = true;

init();
animate();

function init() {
  if (!fbxUrl) {
    return false;
  }
  container = document.createElement( 'div' );
  //创建div，并加载到html里，这里的document.body可以换成你想让模型加载的地方。
  document.body.appendChild( container );
  //创建场景
  scene = new THREE.Scene();
  //3D盒子
  bbox = new THREE.Box3();
  //场景背景颜色
  scene.background = new THREE.Color( 0xeeeeee );
  //半球光
  light = new THREE.HemisphereLight( 0xbbbbff, 0x444422, 1.5 );
  light.position.set( 0, 1, 0 );
  scene.add( light );
  //fbx加载器
  var loader = new THREE.FBXLoader();
  loader.load( fbxUrl, function ( obj ) {
    this.setContent(obj);
    scene.add( obj );
  }, undefined, function ( e ) {
    console.error( e );
  } );
  //创建webgl渲染器
  renderer = new THREE.WebGLRenderer( { antialias: true } );
  renderer.setPixelRatio( window.devicePixelRatio );
  renderer.setSize( window.innerWidth, window.innerHeight );
  renderer.gammaOutput = true;
  container.appendChild( renderer.domElement );
```

```
    window.addEventListener( 'resize', onWindowResize, false );
    camera = new THREE.PerspectiveCamera(45,window.innerWidth / window.innerHeight,    0.
,10000);

    controls = new THREE.OrbitControls(camera);
    // to disable pan
    controls.enablePan = false;
    // to disable zoom
    controls.enableZoom = false;
    controls.target.set(0,0,0);
    controls.update();
}

function onWindowResize() {
    camera.aspect = window.innerWidth / window.innerHeight;
    camera.updateProjectionMatrix();
    renderer.setSize( window.innerWidth, window.innerHeight );
}
//
function animate() {
    requestAnimationFrame( animate );
    if (rotating) {
        scene.rotation.y += -0.005;
    } else {
        scene.rotation.y = scene.rotation.y;
    }

    renderer.render( scene, camera );
}


function setContent(object) {

    object.updateMatrixWorld();
    const box = new THREE.Box3().setFromObject(object);
    const size = box.getSize(new THREE.Vector3()).length();
    const boxSize = box.getSize();
    const center = box.getCenter(new THREE.Vector3());

    object.position.x += object.position.x - center.x;
    object.position.y += object.position.y - center.y;
    object.position.z += object.position.z - center.z;

    this.camera.position.copy(center);
    if (boxSize.x > boxSize.y) {
        this.camera.position.z = boxSize.x * -2.85
    } else {
        this.camera.position.z = boxSize.y * -2.85
    }
    this.camera.lookAt(0, 0, 0);
}
```

浏览器最终效果：



原文链接：VEX-IQ 3D 模型解决方案