Technische Universität Berlin
Fakultät IV
Fachgebiet Regelungssysteme
Projekt Regelungssysteme
SoSe 18

Projektpraktikum Automatisierung

Virtual Satellite for Control Teaching

Tianlun Hu(396676)

August 17, 2018

Tutor
Dr.Ing. Thomas Seel
M.Sc. Daniel Ladig

Contents

1 Introduction . . . 1
2 Design Process . . . 2
  2.1 Build 3D model with Babylon.js . . . 3
  2.2 Build dynamic model in MATLAB/Simulink . . . 6
  2.3 Design of control loop . . . 8
  2.4 Connection with Websocket . . . 8
3 Result of Designment . . . 10
4 Summary . . . 11
5 Reference . . . 12

1 Introduction

Virtual model is a simple and clear way to explain the principles or concepts of abstract theories. It is widely used in different fields and branches these days. This project of automation is aiming to build a virtual demo which could visualized some control theories through practical applications in reality. The control parameters of the visualized system should be adjustable to get different behaviors. In this case the demos could be shown in teaching classes to inspire the students in an intuitive way. By changing the values of parameters in control system and comparing between various visible behaviors the students are supposed to understand the control theory more concisely.

In this part of project, I'm trying to build a virtual satellite model as a demo for control teaching. The height of circuit orbit is taken as controlled variable of the feed back control system and it is controlled by a PID-controller with changing the acceleration of the satellite. The closed control loop can be represented as following:

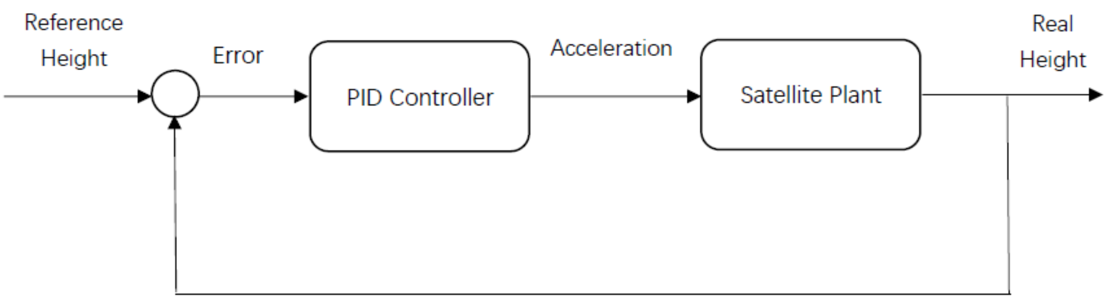

Figure 1: feedback control loop

2 Design Process

In this project, Babylon.js 3D animation engine is introduced as visualized tool. Babylon.js is actually a library of JavaScript based on HTML5. With Babylon.js the demo and animation will be built with JavaScript code and then shown in the interface of browser, which don't need to add any other softwares or change environmental configurations. It means that the users only need a browser to run the demo.

Besides the interface of animations on website, the physical dynamic plant of satellite and relevant control loop should also be implemented for this demo. For this controller and system dynamic part, the MATLAB/Simulink should be introduced. In Simulink the satellite dynamic block is built to simulate the physical behavior of the satellite moving around earth in a circuit orbit. The output of the satellite plant as current position and velocity would be taken by JavaScript and express it in browser. The system model in Simulink will also get the reference height and other parameters(e.g parameters of controller) as input from the JavaScript.

The communication between MATLAB/Simulink and JavaScript would be achieved by a block called WebSocket which is packaged in TUB-control library. With the connection of JavaScript and Simulink model, the data could be transfered simultaneously when user run the demo on website. The block diagram of the tools in this project is shown below.

Figure 2: Tools chain of this project

2.1 Build 3D model with Babylon.js

There are plenty of 3D-models as examples in the Babylon.js library could be used as example. For the first step of this project, a visualization of satellite model should be built with JavaScript in HTML5 files.

The structure of website is set as required, which is separated into three blocks, this is set by the <head> term like the codes shown in Figure3 below. The upper side of the whole website shows the transfer function of control system. The main animation is shown in the block at left. On the right side there are some buttons and diagram of control concept. The detail of the buttons and input box are set inside <body> term

```
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <title>Satellite</title>
    <link href="lib/3rdparty/bootstrap/css/bootstrap.min.css" rel="stylesheet">
    <link href="lib/3rdparty/bootstrap-slider.css" rel="stylesheet">
    <link href="lib/style.css" rel="stylesheet">
    <style>
...
        #topbar {
            width: 100%;
            height: 15%;
        }
        #renderCanvas {
            width: 65%;
            height: 85%;
            touch-action: none;
        }
        #sidebar {
            width: 35%;
            height: 85%;
            float: right;
            overflow: auto;
        }
    </style>
</head>
```

Figure 3: Structure of the Website

The modeling of Satellite 3D model and animations are achieved by Babylon.js with JavaScript inside <script> term. This project task is supposed to build a model that could represent the motivation of controlled satellite. So at first I model a earth as a sphere with texture and add rotate action to modify autobiographic of earth. Then I also modify the galaxy by adding a SkyBox from Babylon.js to make the demo more real, the scale of this SkyBox should be large enough regards to the height of satellite orbit. Part of codes are shown below in Figure4.

```
        //Earth
        var earth = BABYLON.MeshBuilder.CreateSphere("earth", {diameter:30}, scene);
        var earthSurface = new BABYLON.StandardMaterial("earthSurface", scene);
        //earthSurface.diffuseTexture = new BABYLON.Texture("/textures/3_no_ice_clouds_16k.jpg", scene);
        earthSurface.diffuseTexture = new BABYLON.Texture("http://2.bp.blogspot.com/-
        ve1f2PClvus/UGNo6CmITpI/AAAAAAAAANw/IEjDJSmaABQ/s1600/World-satellite+map.jpg", scene, false,
        false);
        earthSurface.diffuseTexture.hasAlpha = true;
        earthSurface.backFaceCulling = true;
        earthSurface.diffuseTexture.uAng = 0;
        earthSurface.diffuseTexture.vAng = 0;
        earthSurface.diffuseTexture.wAng = 0;
        earthSurface.diffuseTexture.uScale = 1.0;
        earthSurface.diffuseTexture.vScale = 1.0;
        earthSurface.diffuseTexture.wScale = 1.0;
        earth.material = earthSurface;

        //Skybox
        var path = "https://raw.githubusercontent.com/BabylonJS/Website/master/Assets/skybox/";
        var skybox = BABYLON.Mesh.CreateBox("skyBox", 1000.0, scene);
        var skyboxMaterial = new BABYLON.StandardMaterial("skyBox", scene);
        skyboxMaterial.backFaceCulling = false;
        skyboxMaterial.disableLighting = true;

        skybox.infiniteDistance = true;
        skyboxMaterial.reflectionTexture = new BABYLON.CubeTexture(path+"nebula", scene);
        skyboxMaterial.reflectionTexture.coordinatesMode = BABYLON.Texture.SKYBOX_MODE;
        skybox.material = skyboxMaterial;
```

Figure 4: Model of earth and SkyBox

After modeling the environment of this satellite demo, the main task is to modify the satellite model and animations of it. The motivation and behavior of the satellite is modified by the dynamic block in Simulink, in order to settle the communication between website and Simulink with Websocket, the 3D model of satellite should be packaged as a class so the data from MATLAB/Simulink could be received by addsample object in class. Part of codes are shown below in Figure5. The 3D model of satellite could be built as a box with two sailboards and two nozzles for accelerating or decelerating. The informations about the status of satellite like position, velocity, acceleration and orientation are all calculated in Simulink dynamic block.

```
class real_Satellite {
    constructor(scene) {
        this.scene = scene;
        this.satelliteM = this.createSatellite();
        this.sateSpeed = this.createSpeed();
        this.line = [];
        }

    createSatellite() {
        const scene = this.scene;
        var OneSatellite = BABYLON.MeshBuilder.CreateBox("OneSatellite", {}, scene);

        OneSatellite.position.z = 36;
        OneSatellite.position.x = 0;

        var Meterial_sate = new BABYLON.StandardMaterial("Meterial_sate", scene);
        Meterial_sate.diffuseColor = new BABYLON.Color3(0.3, 0.95, 0.2);
        OneSatellite.material = Meterial_sate;
```

Figure 5: Model of Satellite

Besides the controlled satellite, another transparency satellite with the same size as controlled one is built as reference. It means that the reference satellite will indicate the reference states for the controlled satellite when the orbit height is changed. This reference satellite has no nozzles because it will jump to the reference height directly without any thrusting action. The dynamic behavior of reference satellite is also included in Simulink model.

Eventually the satellite model would be called in main HTML file to show the complete interface of the website. The final performance in browser is shown in Figure6.

Additionally, in order to instruct the demo more clearly and attractive, some animations are necessary when the satellite is taking actions like pushing and falling to earth. These functions are also packaged inside the class of satellite and would be called in main code under the specific situation.

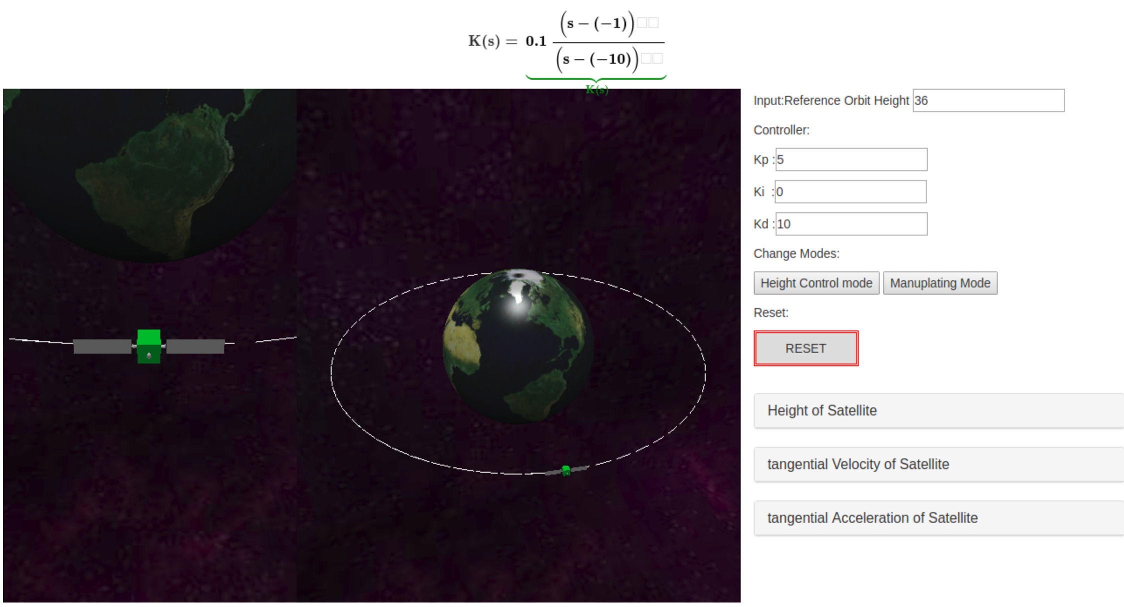

Figure 6: Interface of website

2.2 Build dynamic model in MATLAB/Simulink

After the 3D model building of the whole demo is finished, the control loop and dynamic part of satellite motivation should also be modified. In this project MATLAB/Simulink is used as simulation tool, and the Simulnk model must include a WebSocket block for data interchange with Web page. This block has been installed in Tool-Chain of this project.

For the simulation of satellite behavior, in this task I simply assume that the satellite is moving around earth in a circle orbit and the change of orbit height is happening only in a plane. In this case the change of the orbit could be obviously suggests the control behavior without other side reaction.

The motivation of satellite around the earth as circular motion should satisfy the real physical relation, i.e law of universal gravitation. The centripetal force which refers to the tangential speed of satellite is provided by gravitation, and the gravitation is also related to the height of satellite. This relation could be represented with the equation below:

$$m \cdot \frac{V_t^2}{R} = G \cdot \frac{M \cdot m}{R^2} \tag{1}$$

Here M is quality of earth, m is quality of satellite, G is the gravitational constant, R is the distance from satellite to the geocentric and V_t is tangential velocity of satellite.

In reality, if satellite need to change the height of orbit, it need to change the tangential velocity with accelerating or decelerating by nozzles. Once the tangential speed is changed, the balance of the equation above would be broken, the only way to get steady sate again is changing the value of R. This equation is the basic principle of satellite orbit change.

However, during the modeling of this dynamic behavior I found that if I take the tangential acceleration as controlled variable, this block of satellite plant is nonlinear and it will be much more complex to linearize it.

So in this case I assume that the satellite could apply a acceleration in radial direction and take this radial acceleration as controlled variable. Then the tangential speed would change with the radius of the circle orbit. Then the structure of satellite dynamic model in Simulink is suggested in Figure7. In Babylon.js the default coordinate system is Cartesian coordinate system, but in Simulink model the Polar coordinate system is used to simplify the calculation process.

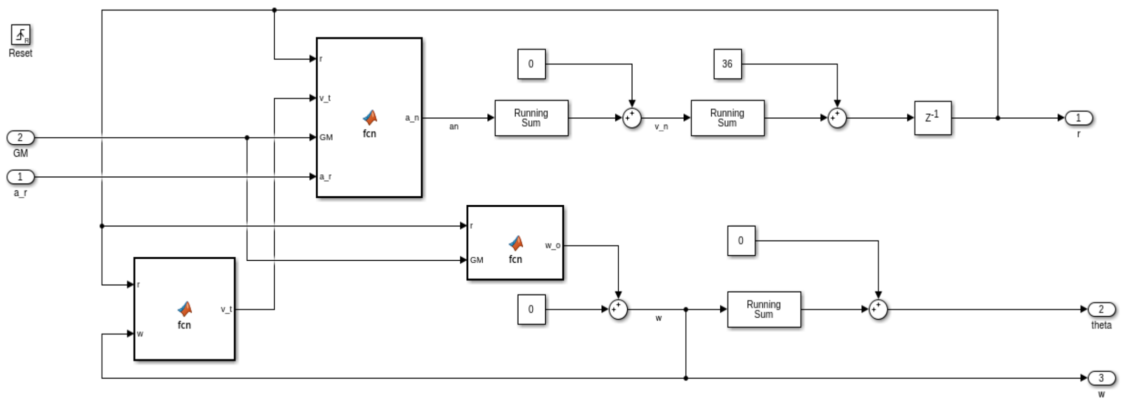

Figure 7: Satellite dynamic model

In order to simulate the real behavior of satellite orbit change, another Simulink block is implemented. Since the real satellite dynamic model is nonlinear, it can't be controlled with a linear controller. In this situation a manipulating mode is introduced, the tangential acceleration could be given with external input device like keyboard or IMU. When satellite gains a acceleration on a circle orbit, it won't turn to another circle orbit but drive into a ellipse orbit instead under the effect of gravitation. This part of Simulink model could modify that motivation, the structure is shown in Figure8. For the sake of simplifying calculation this model is using Cartesian coordinate system.

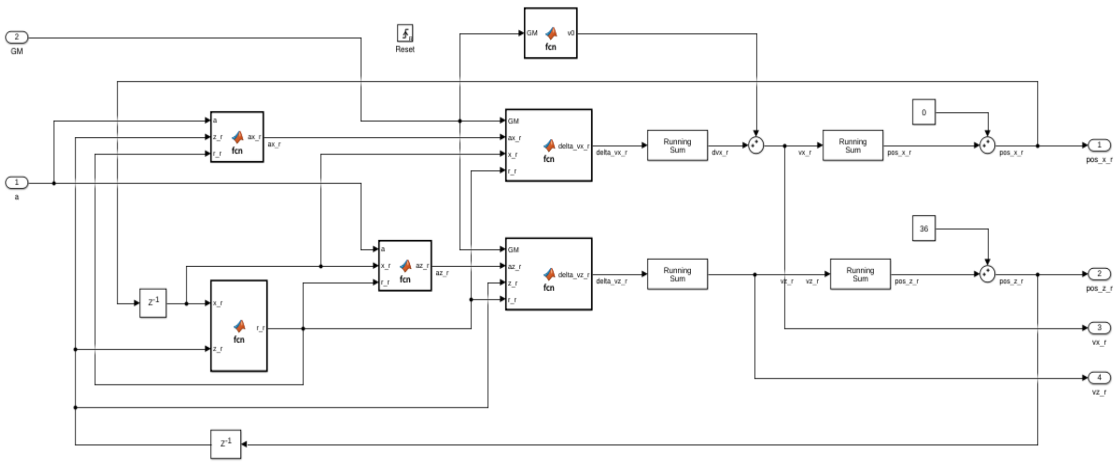

Figure 8: manipulating satellite model

2.3 Design of control loop

For the control part of satellite, the control variable is the height of satellite, i.e this radius of circle orbit. The PID controller take the error between reference height and the current height of satellite as input, the output of controller is radial acceleration as control signal. So in this case only proportional and differential terms of controller is applied, i.e the type of controller is PD.

Then the satellite plant take radial acceleration as input to adjust the height of satellite. After a feedback loop the current height of satellite is taken to compare with the reference value.

For the sake of demonstrating different behavior of control loop with different parameters, this PID controller is built with adjustable control parameters. The structure of control loop in Simulink is shown below in Figure9.

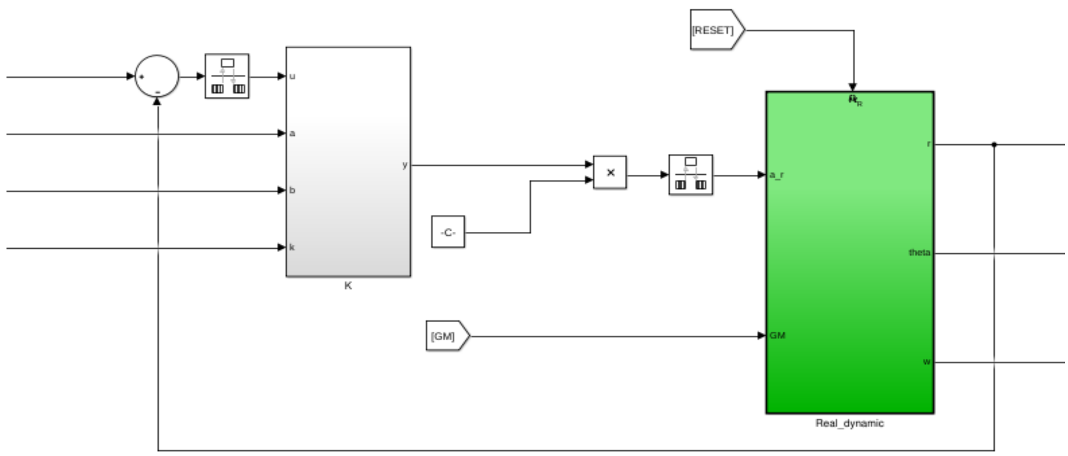

Figure 9: control loop model

2.4 Connection with Websocket

After the designment of visualization 3D model and dynamic part of satellite, in the next step these two models need to be connected for information exchange. In this project Websocket is introduced as communication tool, which could initialize Simulink model on the output side and return the values that calculated by Simulink on the input side.

Besides these main blocks in Simulink model, there are also some other terms for the realization of functions like changing modes and reset. Eventually, the entire model in Simulink is suggested in Figure10 here below.

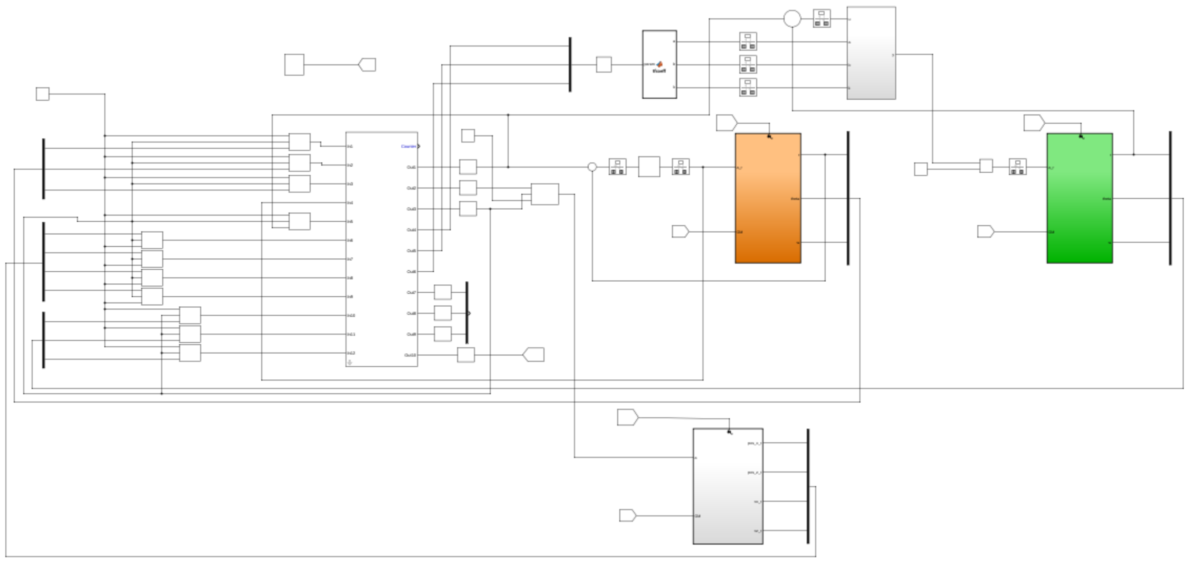

Figure 10: structure of Simulink model

3 Result of Designment

Now the design process is finished, after the implement of Simulink model and run the generated executable file the physical model of demo could be started. This model would be called in HTML file and Chromium browser will present the 3D visualization in webpage. The final result of running demo is shown in the Figure11.

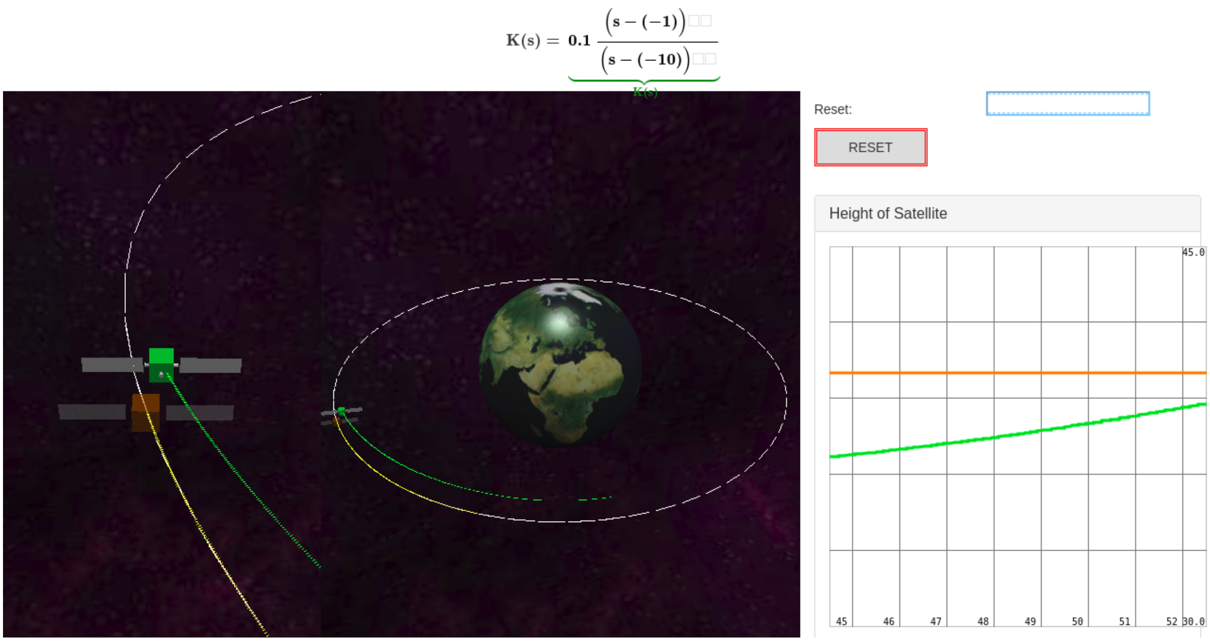

Figure 11: Final result

After a serious of test the satellite demo works well but not flawlessly. The basic function of the satellite demo has been achieved but for now the controller still not work optimally. In the most situation it will take relative long time to reach the next steady state.

4 Summary

This satellite with adjustable controller could be helpful for students in some basic control teaching classes to understand the feedback control and function of PID controller directly. It is meaningful to show these abstract control theories by animations.

About the future works, some control concept like Bode diagram or Root locus can be applied to this demo so the usage of this demo would be developed.

5 Reference

[1] Jun Kyu Lim, Won Hee Lee and Chan Gook Park. Simulink-Based FDI Simulator for Autonomous Low Earth Orbit Satellite[J]. Proceedings of the 18th World Congress The International Federation of Automatic Control Milano (Italy) August 28 - September 2, 2011

[2] Li Jisu, Mu Xiaogang and Zhang Jinjiang. (2004). Physical Simulation for Satellite Control Systems[J]. Beijing Institute of Control Engineering. Aerospace Control, 22, 38-45.