

Design and Implementation of Software Planning System for Dental Robot

Lihua Yang^{1,a}, Chuanhong Zhou^{1,b} and Tao Zhou^{1,c}

¹School of Mechatronic Engineering and Automation, Shanghai Key Laboratory of Intelligent Manufacturing and Robotics Shanghai University, Shanghai, China

Email: ^a871218424@qq.com; ^b13817297415@139.com; ^c1358533088@qq.com

Abstract: With the improvement of people's living standards and the aging of the population, the demand for the dental restoration market continues to increase. However, due to the low efficiency and high prices of traditional technologies, it cannot meet the needs of most people. The dental robot can make up for the above deficiencies. The dental robot is mainly divided into two systems, namely the software planning system and the robot body and control system. This article mainly uses the secondary development function of three-dimensional animation software Blender and the knowledge of prosthodontics to research and develop a CAD software system (a software planning system for dental robots) for dental implants.

Keywords: dental implant, dental robot, secondary development

1. Introduction

With the improvement of people's living standards, changes in eating habits and structure, and aging population, the demand for dental restoration market continues to increase. However, due to traditional technologies and backward consumption concepts, China's oral remediation and cultivation market is still relatively small. Traditional restorations cannot meet people's increasing needs in terms of technology, materials, aesthetics, and price. CAD/CAM technology has penetrated almost all areas of engineering technology and human life. One of the most typical examples of this technology in the field of oral medicine is the pioneering introduction of the design and manufacture of oral fixed prostheses [1].

Modern dental prosthetic technology is combined with the secondary development technology of three-dimensional animation software Blender. By constructing a three-dimensional CAD system for dental restoration, the dental implant process is modeled in three-dimensional simulation, and the simulated data is transmitted to in the robot system. The robot implants new teeth into the patient's mouth based on the acquired data and pre-programmed procedures. This software system can precisely plan the planting plan according to the patient's oral structure, and achieve full navigation and autonomous control during the operation.

2. The Overall Structure of the Dental Robot Software System

The software system of the dental robot researched and developed in the article adopts standard dental data and standard implant data methods. That is based on standard dental implants and standard implants. The CT scan model of the patient's oral cavity is introduced, according to its specific basics. The characteristic structure is to quantify the standard teeth and standard implants and then re-simulate the implant process on the patient's oral model until the implants and implants that match the patient's mouth are generated. The shape of dental implants was modified using Blender's Hook modifier and



Laplacian modifier, combined with various methods of surface creation to reconstruct the dental implant's data structure model.

This software system is composed of four modules: the oral implant user interface module, the dental implant and implant module developed using the Python language, the auxiliary tool module, and the communication module with the robot part.

The software system controlling panel user interface developed is shown in Figure 1. For the system function of oral dentistry, the control panel is designed as five main menus: Teeth, Implants, Bridges, Splints, Ortho. The five main functions are set as buttons on the control panel. When the button is pressed, the corresponding drop-down bar below will show the content. The operation flow of the software system is:

The first step: A CT scan is performed on the oral cavity of a patient who needs to implant a tooth to obtain a patient oral data model, and then the resulting data model is imported into the software system.

The second step: According to the imported patient oral data model, select the appropriate dental data model and implant data model, then perform three-dimensional simulation of the entire planting process, and modify the initial selected implant and tooth shape data to achieve the best position and depth of planting.

The third step: The three-dimensional simulation data will be saved and transmitted to the robot system. After the simulation the patient oral data model is saved as an STL format file.

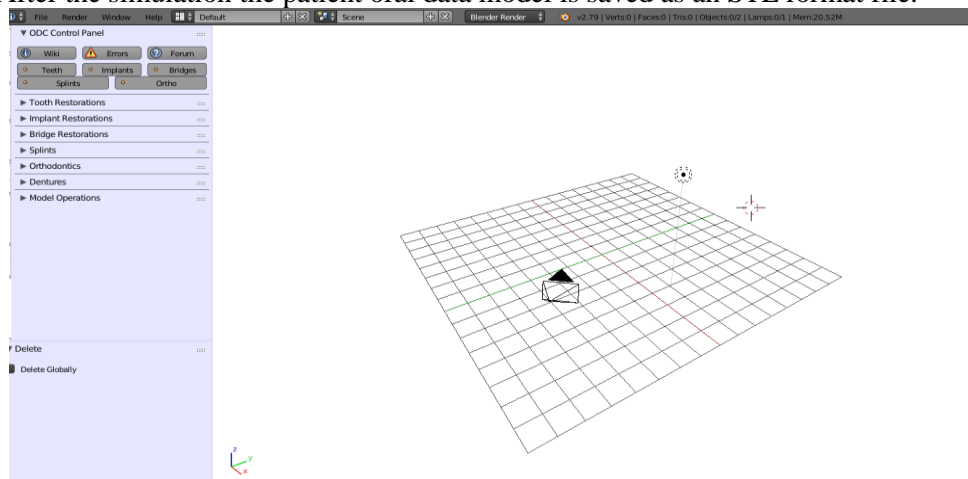


Figure 1. software system interface

Due to the different design of dental implants and the design of industrial products, the design of industrial products is based on a uniform design standard, and this software system is based on the patient's oral form. The software system structure is shown in Figure 2.

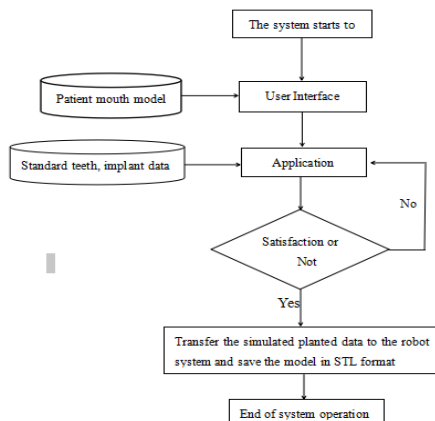


Figure 2. System Structure

```

import bpy
class SimpleOperator(bpy.types.Operator):
    bl_idname = "object.simple_operator"
    bl_label = "Tool Name"
    def execute(self, context):
        print("Hello World")
        return {'FINISHED'}
def register():
    bpy.utils.register_class(SimpleOperator)
def unregister():
    bpy.utils.unregister_class(SimpleOperator)
if __name__ == "__main__":
    register()
  
```

Figure 3. Python script

3. Key Technologies for Implementing Dental Robot Software System

3.1. Blender Secondary Development Technology

Blender provides very powerful API support. Because Blender is built on the Python language. Its API is fully compatible with the Python language module, so it has a very strong multimedia application scalability [2]. Blender embeds a Python interpreter that is started and active by Blender. This interpreter runs scripts to draw the user interface and is used for some of Blender's internal tools. Blender provides a bpy module for the Python interpreter that can import this module into a script and provide access to Blender data, classes and functions. In order to interact with Blender, scripts need to use tightly integrated APIs. Blender provides a lot of interface functions for secondary development of the Python language.

Blender provides a Python API, which provides a platform for deep interaction between Blender and Python. Users can use standard or custom Python modules to invoke Blender's default Python library, which will give Python its power [3]. The Blender Python API provides the following major modules for manipulating objects:

- (1) bpy.ops.object: Selected operation object.
- (2) bpy.data: Operation of the currently open Blender file data.
- (3) bpy.context: Operates the text content of the selected object.

The secondary development Python script based on Blender is mainly composed as follows Figure 3: First, defining a subclass of bpy.types, both class attributes begin with the bl_ prefix. This is a convention used to distinguish Blender-defined variables from your own. The Execute function receives an operator and an instance of the current context. Finally, the registration function is called and it loads the class into Blender. The Blender module loaded at startup needs the register() and unregister() functions. These are the only functions that can be called in your code, otherwise it is a regular Python module.

3.2 Design of Dental Model Library

Since the standard tooth data determines the basic morphology of the outer surface of the implanted tooth, the acquisition of standard dental data is also crucial. The use of conventional methods, such as mechanical scanning or laser scanning, cannot obtain complete and accurate surface data for individual teeth. The reason is that it is difficult to obtain surface data for areas below the high point of the crown profile in the previous scanning methods. With the successful development of a new generation of 3D laser scanners, we have a new means to obtain data [4]. Therefore, we used a 3D dental model laser scanner [5]. to scan a superhard plaster model of 28 standard crowns/bridges provided by the hospital as 1:1. Through the scanner scan, we obtained the full dentition of surface 3D point cloud data of 28 standard dental crowns [6]. The completed standard dental model library is shown in Figure 4.



Figure 4. Standard Dental Model Library

The module code of the dental system to retrieve dental data from the standard dental model library is shown in Figure 5:

```

class CBGetCrownForm(bpy.types.Operator):
    '''Inserts a crown form from the tooth library at the cursor location'''
    bl_idname = 'opendental.get_crown_form'
    bl_label = "Get Crown Form"
    bl_options = {'REGISTER', 'UNDO'}
    bl_property = "ob_list"

    def item_cb(self, context):
        return [(obj.name, obj.name, '') for obj in self.objs]

    objs = bpy.props.CollectionProperty(type=bpy.types.PropertyGroup)
    ob_list = bpy.props.EnumProperty(name="Tooth Library Objects", description="A List of the tooth library", items=item_cb)

    def invoke(self, context, event):
        self.objs.clear()
        settings = get_settings()
        #here we grab the asset library from the addon prefs
        libpath = settings.tooth_lib

        #a list of all objects in the asset library
        assets = odcutils.obj_list_from_lib(libpath, exclude = '_')
        for asset_object_name in assets:
            self.objs.add().name = asset_object_name
        #context.window_manager.invoke_search_popup(self.ob_list)
        context.window_manager.invoke_search_popup(self)
        return {'FINISHED'}

```

Figure 5. To retrieve the tooth code from the standard tooth library



Figure 6. Standard implant model library

3.3 Design of the Implant Model Library

Implants are generally made of artificial materials (such as metal, ceramics, etc.), and threaded columnar and root-shaped implants have become widely accepted implant forms worldwide. At present, implants with threaded roots and parallel-wall designs are the mainstream of clinical application [7]. Therefore, we also used a three-dimensional laser scanner to scan the standard implants provided by the hospital to obtain three-dimensional point cloud data to form a standard implant model library. The completed standard implant model library is shown in Figure 6:

The module code for calling the implant from the standard implant library in the software system is shown in Figure 7.

4. Software System Operation and Results

After a series of compiled Python code packages are compressed and installed on the Blender in the form of plug-ins, the entire software system interface will be generated. Firstly, after scanning the patient's oral three-dimensional model into the software system (Figure 8a), and then import the implant and import standard dental implants (Figure 8b and Figure 8c), for oral dental implant simulation (Figure 8d). Finally obtain the desired data and pass the data to the robot system.

```

class OPENDENTAL_OT_place_implant(bpy.types.Operator):
    '''Places Implant or swaps existing implant with new implant of your choice'''
    bl_idname = "opendental.place_implant"
    bl_label = "Place Implant"
    bl_options = {'REGISTER', 'UNDO'}
    bl_property = "imp"

    def item_cb(self, context):
        return [(obj.name, obj.name, '') for obj in self.objs]

    objs = bpy.props.CollectionProperty(type=bpy.types.PropertyGroup)

    imp = bpy.props.EnumProperty(name="Implant Library Objects",
                                description="A List of the tooth library",
                                items=item_cb)

    hardware = bpy.props.BoolProperty(name="Include Hardware", default=False)

    @classmethod
    def poll(cls, context):
        return len(context.scene.odc_implants) > 0

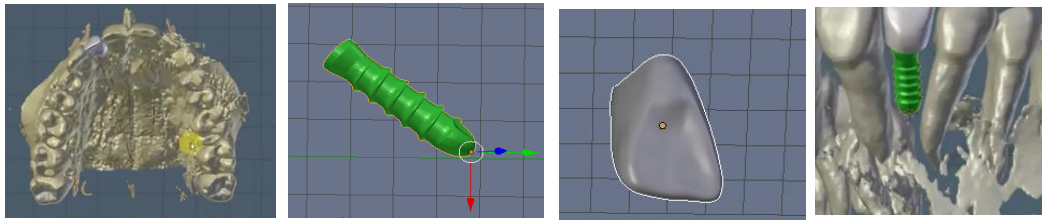
    def invoke(self, context, event):
        self.objs.clear()
        settings = get_settings()
        libpath = settings.imp_lib
        assets = odcutils.obj_list_from_lib(libpath, exclude = '_')

        for asset_object_name in assets:
            self.objs.add().name = asset_object_name

        context.window_manager.invoke_search_popup(self)
        return {'FINISHED'}

```

Figure 7. Calling the implant module code diagram



(a) Introduce the patient's oral three-dimensional model

(b) Import the implant

(c) Import standard dental implants

(d) Conduct dental implant simulation

Figure 8. Three-dimensional simulation and simulation process diagram of dental robot software system

5. Communication Between Software System and Robot System

The robot system uses ABB's robots--model IRB 140TW. IRB series general-purpose robots are the main products of ABB robots. This series of robots is a very mature 6-axis industrial robot. This series of robots is an open-architecture design to adapt to different occasions, and can easily communicate with external systems [8]. Because the software system communicates with the robot system, the simulated data is transmitted to the robot system, and ABB robot provides a rich I/O communication mechanism, which can easily communicate with the PC and surrounding equipment [9]. table 1 shows the communication method of ABB robots. Therefore, the Socket message communication method is adopted between the software system and the robot system, and the main mode of connection-based Socket communication is the client/server method [10]. Among them, the PC acts as a server and the ABB robot acts as a client.

Table 1. ABB robot communication

PC	Field bus	ABB standard
RS232	Device	Standard I/O edition
OPC server	Profibus	PLC
Socket Message	Profibus-DP
	Profinet
	EtherNet IP

The code module using Socket message communication between the software system and the robot system is shown in Figure 9.

```

TCP_IP = '127.0.0.1'
TCP_PORT = 5005
SOCKET = None
RAYPUMP_PATH = None
class MessageRenderOperator(bpy.types.Operator):
    bl_idname = "object.raypump_message_operator"
    bl_label = "Save & Send To data"
    bl_description = "Saves, sends and schedules current scene to Robot system"

    def connect(self):
        global SOCKET, RAYPUMP_PATH, TCP_IP, TCP_PORT
        print("New connection!")
        if SOCKET != None:
            SOCKET.close
        # Create a connection:
        try:
            SOCKET = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        except socket.error as msg:
            self.report({'ERROR'}, "Cannot create a socket")
            SOCKET = None
            return {'CANCELLED'}

```

Figure 9. Code module for communication between software system and robot system

6. Conclusions

This article adopts Blender's secondary development technology to develop a software system for dental robots. This system is mainly aimed at oral dental implants. The main target is dentists. The software system has a very good user interface and can easily import models required for various dental implantation procedures. It is very easy to learn and at a glance. The development of this software system has solved the domestic traditional oral dental implant methods in the past, greatly improving the precision and speeding of dental implant surgery, and making the whole dental implant surgery process quick and accurate.

Acknowledgment

We thank Shanghai Key Laboratory of Intelligent Manufacturing and Robotics for assistance with our work.

Reference

- [1] Wang Shujie, Lin Ying, Song Lina. Orthopaedics fixing CAD system based on surfacer secondary development[J]. Computer Engineering and Applications, 2004(28):191-194.
- [2] Hao Zhenhua. Research on dynamic simulation application of python in Blender engine[J]. Software Guide, 2012, 11(11):66-68.
- [3] Hao Zhenhua. Research on dynamic simulation application of python in Blender engine[J]. Software Guide, 2012, 11(11):66-68.
- [4] Zou Bo, Lu Peijun, Wang Yong. Acquisition of standard crown data [J]. Journal of Practical Stomatology, 2002(06): 550-552.
- [5] Zou Bo, Lv Peijun, Wang Yong. A comparative study on the reliability of a new three-dimensional dental model laser scanner. Journal of Practical Stomatology, 2002,18 (6): 546
- [6] Cheng Yusheng, Dai Ning, Liao Wenhe, Yu Qing. Establishment of standard crown and bridge database[J]. Journal of Southeast University(Medical Science Edition), 2007(01):1-3.
- [7] Huang Yuanliang. Development trend of dental implant materials and morphology design[J]. Journal of Dental Materials and Devices, 2015, 24(03): 113-117.
- [8] Deng Sihao. Research and implementation of real-time robot monitoring and communication system[D]. Wuhan University of Technology, 2003.
- [9] Sun Zhiwei. Research on ABB industrial robot network integrated control system[D]. Yanshan University, 2017.
- [10] Zhao Y. Research and implementation of message queue middleware based on Socket [D]. Inner Mongolia University, 2007.