Pepr3D

Authors: Bc. Štěpán Hojdar, Bc. Tomáš Iser, Bc. Jindřich Pikora, Bc. Luis Sanchez
Supervisor: Mgr. Oskár Elek, PhD.
Consultants: doc. Ing. Jaroslav Křivánek, Ph.D., Ing. Vojtěch Bubník (Prusa Research s.r.o.)

Faculty of Mathematics and Physics
Charles University

Contents

Chapter 1

Introduction

1.1 3D printing basics

3D printing is a new technology that has seen rapid developement in the last years. It comes in many different forms, melting plastic, fusing metals, shining UV on photopolymers, etc. Fused Deposition Modelling (FDM) is the most popular and accessible to the general public and for the purpose of this project, when we talk about 3D printing, we will always mean FDM printers, unless stated otherwise.

FDM printing is a relatively simple process - a printer head melts the plastic filament and deposits it on a preheated platform layer by layer, from the bottom towards the top. The printer has to regulate the temperature of both the filament in the head and the moving platform for the deposited material to bond correctly. Several types of filaments are used, namely PLA, ABS, PET and others.

1.2 Prusa environment

The Prusa environment is very similar to the general description we provided in the section 1.1. For the puprose of our project, the most important concept in the Prusa environment is the slicer. The slicer is a program that receives the 3d model the user wishes to print out and creates the instructions for the Prusa 3d Printer - a G-code file. The file is then transfered to the printer, which then executes the commands in the G-code file. The slicer has to plan the movement of the head for the whole print. This includes several crucial things:

- Covering the whole area of each layer
- Reinforcing the walls of the object to make them sturdier
- Filling the inside of the object with a rougher print, because it won't be visible when finished
- Planning the path so the head can stay in one Z level - an "Eulerian path".
- Switching the materials for multimaterial printing (more in 1.3)

Prusa develop their own slicer - a forked branch of an open-source program called Slic3r [1], called Slic3r Prusa Edition [2]. This slicer can do all we listed above very well.

1.3 Multimaterial printing

Multimaterial printing is a very new concept, even in the fairly new world of 3D printing. Many of the simpler and cheaper 3D printers can only print one material models - one color for the whole object. However, many users would like to print models that include more than one color. Even though the more advanced printers are capable of combining up to four different materials into one print, the process to achieve this is rather cumbersom for the end user - the user has to manually split the 3D mesh of the object into parts that he wishes to have a different color.

For example, if we are printing a dragon, want the dragon to be black and have white teeth, we have to take the dragon model, and split off each individual tooth. Then tell the slicer that the remaining file - the toothles dragon should be black and the teeth should be white.

This model splitting has to be done in a full 3D editing software like Blender or 3ds Max, which is difficult to control for newcomers and overly complex.

1.4 Our project

Our project aims to make printing a multi-colored object a lot easier, by developing an application that will allow the user to simply paint on the 3D model (i.e. the dragon) with different colors (i.e. color the teeth white), then simply click export and generate the files of the split-off models automatically.

Our application should allow for free hand painting as well as some forms of guided painting - bucket fill and some smarter tools, for example a bucket fill that studies the object's geometry and stops the filling if it detects a sharp edge (i.e. the transition of the tooth into the dragon).

Our aim is to make the application for desktop PCs, with main developement time being focused on the Windows operating system. However, we are trying to use software engineering tools that can also be ported to a plethora of other platforms like Linux based OS, Mac OS and mobile, if the need should arise.

1.5 Related works

Based on the analysis of the experts from Prusa Research s.r.o, there, at the moment, does not exist a software that does what this project is trying to achieve.

The closest existing software is Autodesk Meshmixer [3], which is very complicated and is not targeted for FDM printing specifically. As such, it includes a lot of features that are not important for the FDM users and end up being confusing.

[1] http://slic3r.org/

[2] https://www.prusa3d.com/slic3r-prusa-edition/

[3] http://www.meshmixer.com/

Microsoft 3D Builder [4] is another application that handles 3D models but we have not found a way to make it create anything remotely applicable to FDM printing.

1.6 Challenges in this project

This section should briefly familiarize the reader with some of the parts of the application we think will be difficult to implement correctly, before we present the full program specification.

1.6.1 Handling the geometry during editing

We want our application to be able to emboss text on the surface of the object, detect edges and stop painting the color during bucket fills, allow the user to paint fine details on a rough triangle mesh. All of these things require some degree of subdividing the triangle mesh to allow the user to create small details. We think that this potentially could involve some difficult problems - we have to allow the user to subdivide the triangle mesh enough to actually allow him to create fine details on the surface. However, the if the user goes overboard with the subdivision, the model will be too complex to print or even handle inside a desktop PC.

1.6.2 Exporting the finished objects

After the user is done painting, the application will have to separate the designated objects and areas into distinct meshes. This is potentially a very complicated task to do correctly for non-convex meshes. For example: if we are writing a text on a ball, we really only want the text to be carved deep enough into the ball for the printed material to hold firmly, we do not want it to be too deep. However, if we want the dragon's teeth to be white, we would prefer the whole tooth to be white, not just its surface. The distinction between these two cases could be non-trivial.

1.6.3 Performance

Handling complex geometry is a very taxing task for the user's computer. This application is targeted on beginner-level customers of simple and non-expensive 3D printers. Therefore the application cannot be too hardware demanding - it has to run smoothly on an average 3 year old PC or Notebook. We expect it is going to be hard to ensure this is the case.

[4] https://www.microsoft.com/en-us/p/3d-builder/9wzdncrfj3t6?activetab=pivot%3Aoverviewtab

Chapter 2

Use case

2.1 Wireframe of the application

Figure 2.1 provides a simple wireframe sketch of the application graphical user interface (GUI). The window consists of several usual components:

- A horizontal toolbar at the top, allowing for a fast selection of tools and file manipulation
- A 3D preview window, with live preview of the object. This window allows rotating and magnifying the object, as well as the application of selected tools (e.g. painting with a brush). A simple grid and a 3D cross is provided to ensure the user is always aware of object orientation, as it is important for 3D printing.
- An options window on the right allows the user to customize the settings of the currently selected tool (e.g. selecting the color for a brush).

2.2 Workflow

In this section we describe the intended workflow for a user who has a 3D model he wishes to color and then print on a multicolor FDM printer. This application is intended for users of varying degrees of experience and our goal is to create as easy-to-use application as possible.

2.2.1 Importing the model

First, the user has to import the model he found or created. Clicking on the *import button* creates a dialog, the user selects the 3D model and the model gets loaded. The application should accept at least a few standard formats – namely Wavefront .obj and .stl [1], both of which are widespread and well known among the 3D printing community.

The user should also be able to continue on an already existnig project made earlier with Pepr3D.

[1] https://en.wikipedia.org/wiki/STL_(file_format)

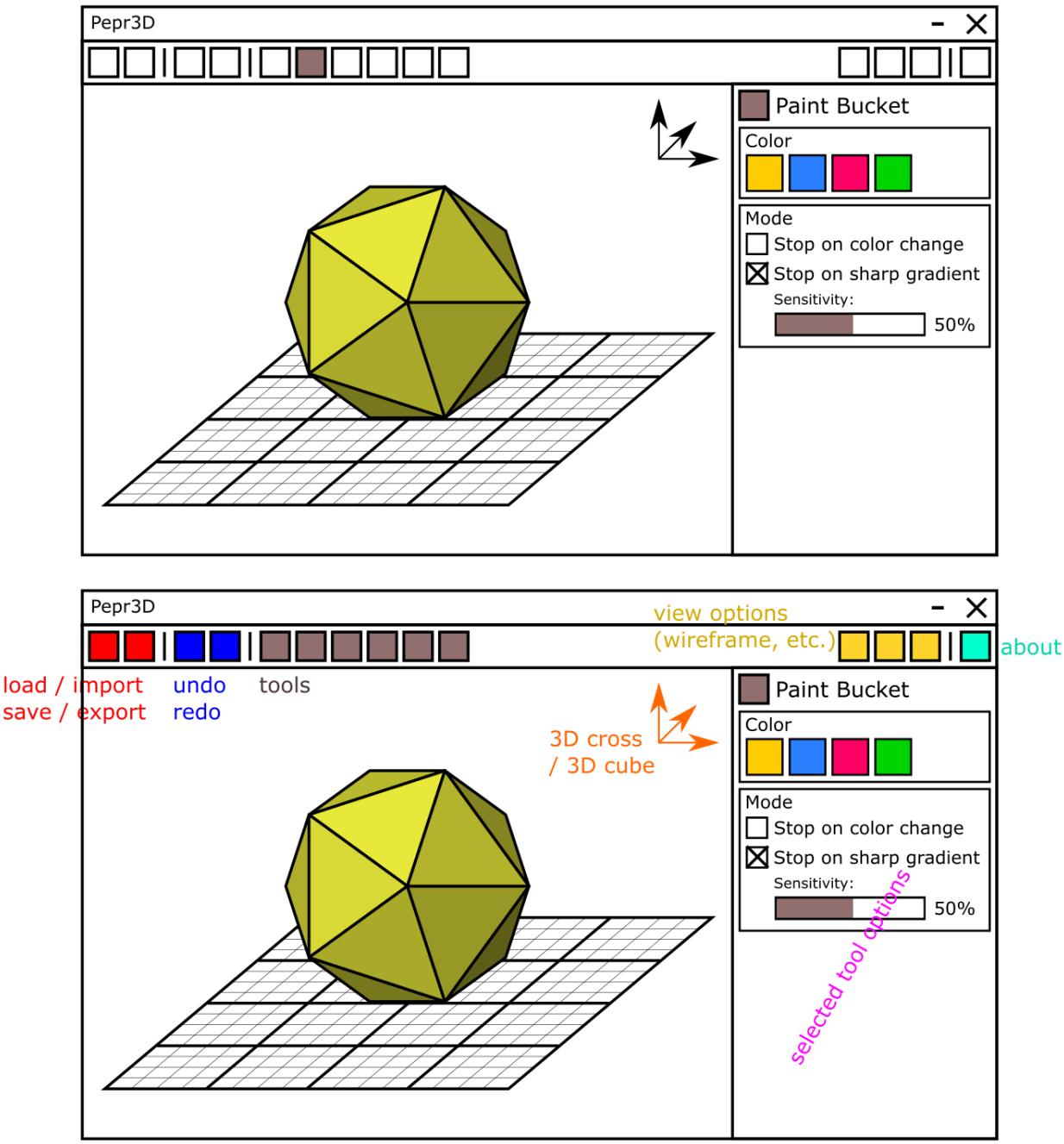

Figure 2.1: A simple wireframe sketch of the application.

After the model is loaded, it will be rendered in the 3D preview window. The window allows the user to rotate, zoom in and out and preview the wireframe of the model (rendering only edges and no faces of the model). The user is able to set the number of colors he wants to use (by default 4 because current Prusa printers support up to four colors). The model is colored with the first color by default.

The user then selects one of the tools from the toolbar. This will bring up the *Tool options* menu on the right hand side allowing the user to customize the tool.

2.2.2 Tools

Edit history

The user is always able to revert his last action by using an *Undo button* or a keyboard shortcut. Depending on the technical difficulties, this feature could also persist through different sessions.

Save as Pepr3D project

Saves the current project as a Pepr3D project file. Upon re-opening Pepr3D, the user can load the project back and continue the work as if he never left. Does not include exporting the file into a slicer-compatible format.

Export

Export the file into a slicer-compatible format. This file is then handed to the slicing program (e.g. Slic3r Prusa Edition we mentioned earlier) and can be printed directly.

Triangle painter

After selecting a color, the user can assign said color to triangles he clicks on. Backside filtering is always on, so the user can only ever color a triangle that faces towards him, which should prevent a lot of accidents a lesser experienced user might make.

Bucket painting

The user selects a color and by clicking anywhere on the model paints all triangles with selected color until an edge criterion is met. The simplest and most intuitive edge criterion is continuity (a hole stops the bucket spread). Several more criterions could be useful when in 3D, namely the sharpness of the normal (if two neighbouring triangles are at an angle greater than X, stop.) or a big gradient in a *shape diameter function* (SDF).

Automatic segmentation

Pepr3D fully automatically colors the whole model using the selected colors, according to a edge criterion as discussed in the *Bucket painting* section. The

Figure 2.2: Semi-automatic segmentation as seen from the user's perspective. The ears of the rabbit are yellow as indicated by one stroke on each ear. The body is orange as indicated by the stroke on its back. The rabbit's feet are pink – four pink strokes.

user can then decide if he wants to merge some segments together, reducing the number of colors.

Semi-automatic segmentation

The user roughly paints over triangles in areas that should have distinct colors, as indicated by Figure 2.2. The program then finishes the coloring by executing a clever flood-fill algorithm utilizing SDF, sharp edges, etc.

Brush

A simple to use brush tool that allows to paint onto the model with a selected color. This tool allows the user to paint finer details, even though the geometry does not include them. For example painting the nose of the rabbit from Figure 2.2 black – there is no distinct edges on the nose, but the user can color only the nose by fine strokes of the brush.

The implementation of this tools is harder, because the program needs to adaptively subsample the triangle mesh to allow for finer details. This poses a lot of problems, which will later be discussed in the implementation parts of the document.

Text

Using the *tool options* window, the user selects a font and types a custom text into a window. The text gets projected onto the model using some sort of projection

Figure 2.3: Three stages of triangle numbers. The bunny on the left has the most triangles and most complicated geometry. Several decimations can reduce the number of triangles but also the number of details as shown on the second and third bunny.

transformation (customizable by the user from a limited range of projections). The software also allows extruding the projected text in the direction of the surface normal to create a 3D effect.

Triangle subdivision/decimation

The user selects a section of triangles and then presses either subdivide or decimate, which will either make the geometry more complicated (and smooth), or simpler and more rough. See Figure 2.3 for visual aid.

Chapter 3

Execution of the project

3.1 Todo by Jindra