Task7&8 纹理图像的创建与编辑

1-0 配置与说明

从 github 网站 https://github.com/swayfreeda/ImageBasedModellingEduV1.0.git 上下载工程文件，里面包含本次课的作业和代码范例(examples/task3)文件夹下。

Clion 软件可以配置 git 版本控制。

将上述工程 fork 到自己的 github 账户下:

首先需要安装 git 软件，

然后从 Clion 菜单 VCS->Git->Clone 进行拷贝。

本节课的代码是在表面重建的基础上进行的，因此需要先运行上次课的代码，生成三维网格。

2-1 运行程序实现点云的清理

运行 examples/task7/class7_mesh_clean.cc，生成三维网格。命令行参数

–t10 –c100 /examples/data/surface_mesh.ply /examples/data/surface_mesh_clean.ply

其中-t10 表示删除最长边和最短遍大于 10 的三角形，-c1000 表示删除顶点个数少于 1000 的独立区域。

2-2 运行程序实现纹理贴图

运行 examples/task7/class7_texturing.cc 生成纹理图像。命令行参数

./examples/data/sequence_scene::undistorted ./examples/data/sequence_scene/surface_mesh_clean.ply ./examples/data/sequence_scene

2-3 推导全局颜色调整的公式

参考 slides-23 全局颜色调整部分的公式以及实例部分，给出公式(1)到公式(2)的推导过程。

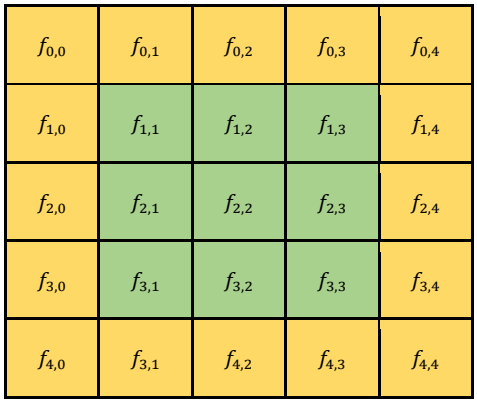
$$\boldsymbol{E}(\boldsymbol{g}) = \sum_{v_i \in seam} \sum_{(l_i^j, l_i^k) \in l_i} \left(\boldsymbol{f}_i^{l_i^j} + \boldsymbol{g}_i^{l_i^j} - (\boldsymbol{f}_i^{l_i^k} + \boldsymbol{g}_i^{l_i^k}) \right)^2 + \boldsymbol{\lambda} \sum_{v_i} \sum_{v_j \in N(v_i)} \sum_{l \in l_i \cap l_j} (\boldsymbol{g}_i^l - \boldsymbol{g}_j^l)^2 \quad \textbf{(1)}$$

$$E(\boldsymbol{g}) = E_s(\boldsymbol{g}) + \lambda E_i(\boldsymbol{g}) = \boldsymbol{g}^T \boldsymbol{A}^T \boldsymbol{A} \boldsymbol{g} + 2\boldsymbol{b}^T \boldsymbol{A} \boldsymbol{g} + \boldsymbol{b}^T \boldsymbol{b} + \frac{\lambda}{2} \boldsymbol{g}^T \boldsymbol{L} \boldsymbol{g} = \boldsymbol{g}^T (\boldsymbol{A}^T \boldsymbol{A} + \frac{\lambda}{2} \boldsymbol{L}) \boldsymbol{g} + 2\boldsymbol{b}^T \boldsymbol{A} \boldsymbol{g} + \boldsymbol{b}^T \boldsymbol{b} \quad \textbf{(2)}$$

全局颜色调整的算法实现在 global_seam_leveling.cc，参考代码验理解推导过程。

2-4 推导泊松表面重建线性方程

slides32 开始部分实例中的泊松数学模型为



$$
\begin{cases}
f_{0,0} = f^*_{0,0} \\
f_{0,1} = f^*_{0,1} \\
f_{0,2} = f^*_{0,2} \\
f_{0,3} = f^*_{0,3} \\
\vdots \\
f_{4,4} = f^*_{4,4} \\
f_{0,1} + f_{1,0} + f_{1,2} + f_{2,1} - 4f_{1,1} = div\boldsymbol{v}(1,1) \\
f_{0,2} + f_{1,1} + f_{1,3} + f_{2,2} - 4f_{1,2} = div\boldsymbol{v}(1,2) \\
f_{0,3} + f_{1,2} + f_{1,4} + f_{2,3} - 4f_{1,3} = div\boldsymbol{v}(1,3) \\
\vdots \\
f_{2,3} + f_{3,2} + f_{3,4} + f_{4,3} - 4f_{3,3} = div\boldsymbol{v}(3,3)
\end{cases}
$$

该线性方程等价写成归一化的形式为

$$\boldsymbol{Lf} = \boldsymbol{b},$$

其中$\boldsymbol{L} \in \boldsymbol{R}^{25\times25}, \boldsymbol{f} \in \boldsymbol{R}^{25\times1}, \boldsymbol{b} \in \boldsymbol{R}^{25\times1}$. 试着将边界像素的变量从向量中移除，以上图为例，$\boldsymbol{f} = [f_{11}, f_{12}, f_{13}, f_{21}, f_{22}, f_{23}, f_{31}, f_{32}, f_{33}]^T$,重新构造方程，给出矩阵表达形式，同时解释其物理意义 (分成三类像素进行解释：边界上的像素，邻域中有像素是边界，邻域中没有像素是边界)。泊松图像编辑的算法实现在 local_seam_leveling.cc 中，参考代码重新理解推导和求解过程。