**Московский авиационный институт**

**(национальный исследовательский университет) «МАИ»**

**Институт №8** **—** «Информационные технологии и прикладная математика»

**Кафедра 806 —** «Вычислительная математика и программирование»

Практическая работа по курсу

**«Процедурная визуализация»**

**Выполнил:**

студент группы М8О-207М

Четвериков В. А.

**Принял:**

Кондратцев В. Л.

Оглавление

[Домашнее задание №1 3](#_Toc61946037)

[Домашнее задание №2 9](#_Toc61946038)

[Домашнее задание №3 11](#_Toc61946039)

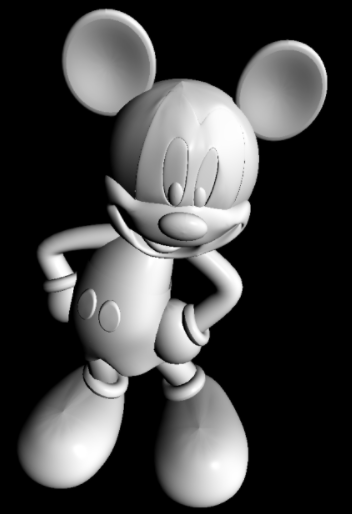
[Домашнее задание №4 14](#_Toc61946040)

[Домашнее задание №5 15](#_Toc61946041)

[Домашнее задание №6 18](#_Toc61946042)

[Приложения 32](#_Toc61946043)

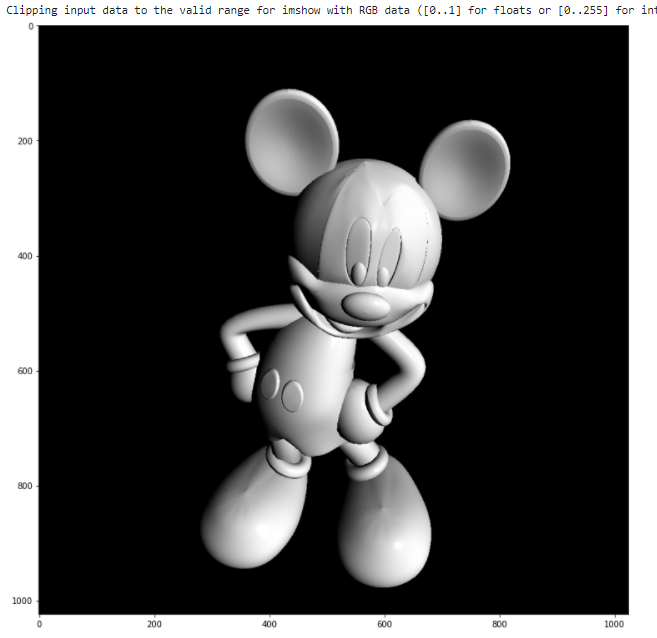
# Домашнее задание №1



*Исходная модель*

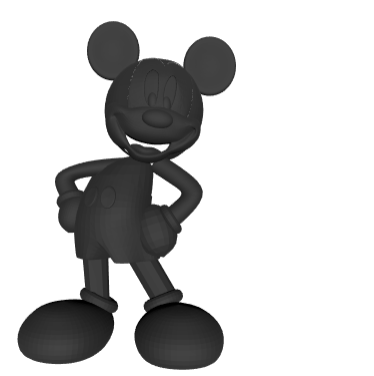
Общее число полигонов: 3 568

Общее число вершин: 6 126



*Рендер модели*

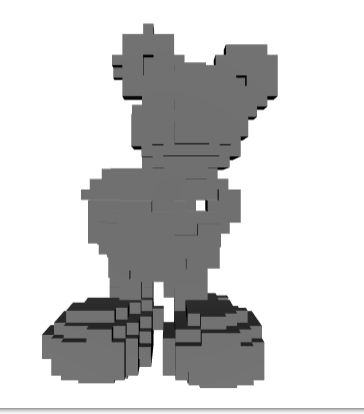
Интерактивная визуализация:

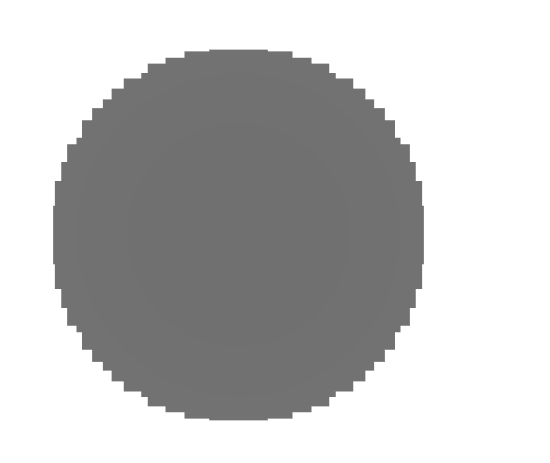


Эйлерова характеристика пространственного графа модели Mickey Xi = V - E + F = 35

Результат вокселизации модели:

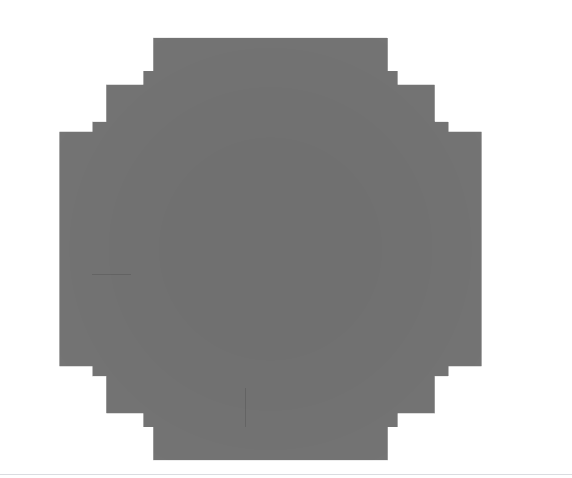
* С высоким разрешением



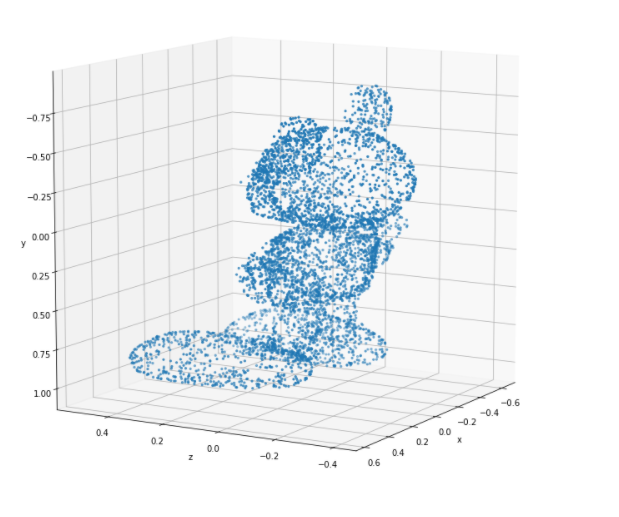


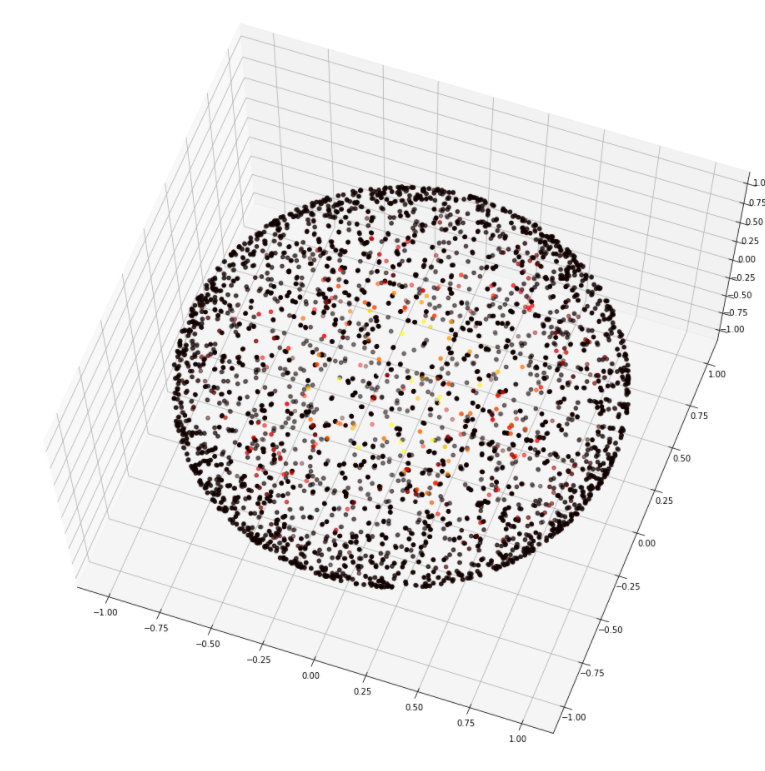
* С низким разрешением





Представление объекта в виде облака точек:





Результат вычисления *SDF*-функции для внутренних и внешних точек рассматриваемой модели:

SDF[-1.82762352e-06 -3.67123677e-05 -1.16962003e-03] = -0.07190296798944473

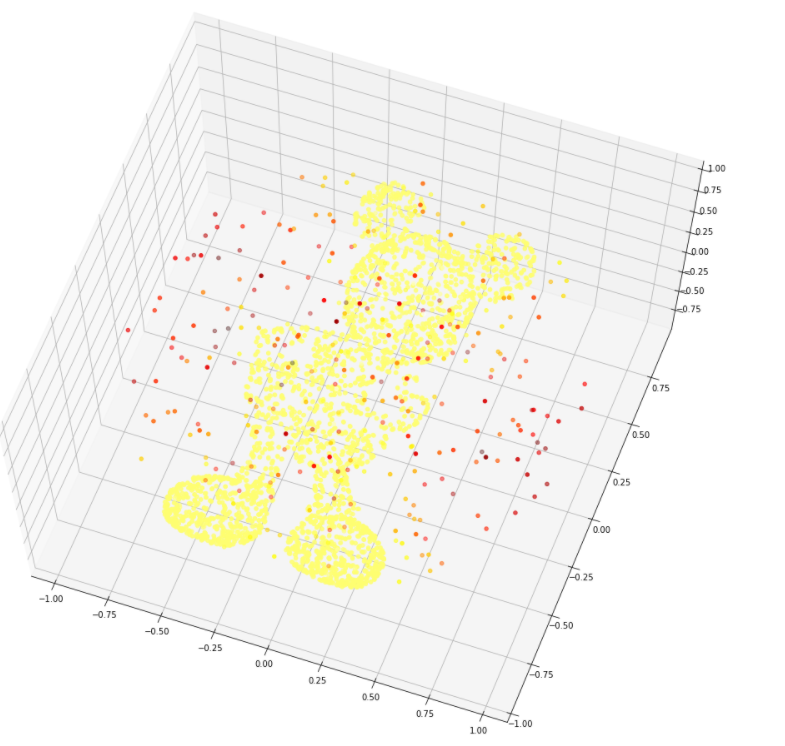
SDF[3. 3. 3.] = 4.590275287628174

Для сферы

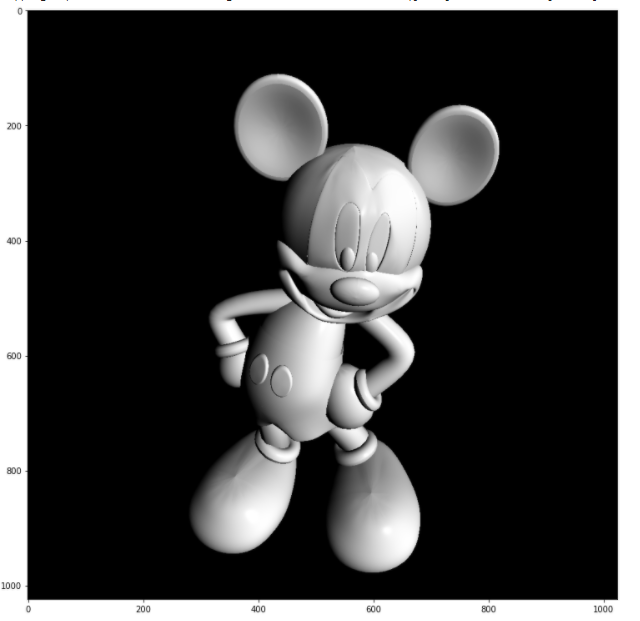
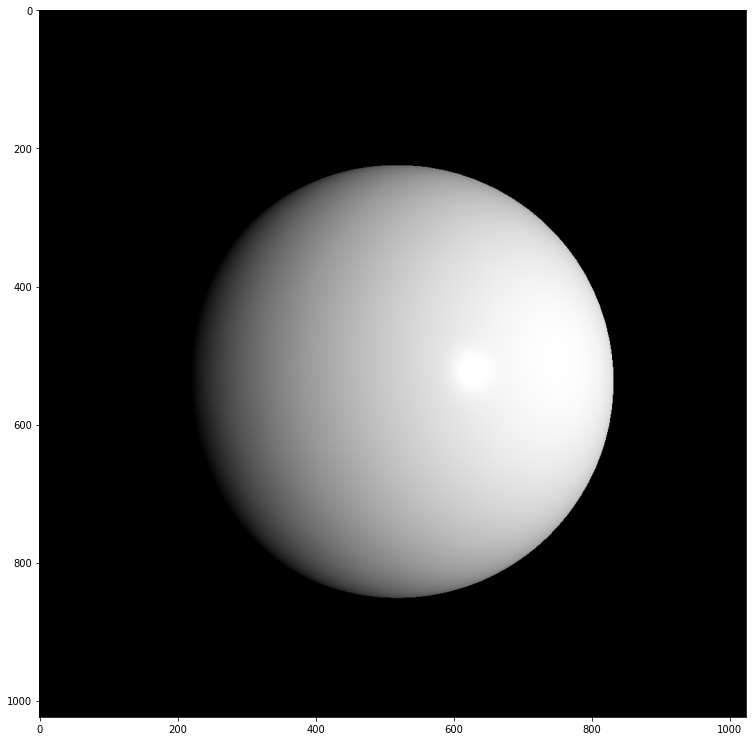
SDF[ 0.0000000e+00 0.0000000e+00 -8.8540061e-18] = -0.998867392539978

SDF[3. 3. 3.] = 4.195400238037109

Визуализция *SDF*, на основе просемплированных точек в окрестности модели:



# Домашнее задание №2



Значения *Chamfer loss* и *Normal loss* для моделей сферы и mickey:

*Chamfer loss = 0.5596358776092529*

*Normal loss = 0.7599741220474243*

Значение регуляризатора формы итогового меша (*shape regularizers*):

*Edge loss for mickey.obj: 0.0005850476445630193*

*Edge loss for sphere.obj: 0.005724199116230011*

Значения регуляризатора *Laplacian loss*:

*Laplacian smoothing objective for mickey.obj: 0.0035534127227990274*

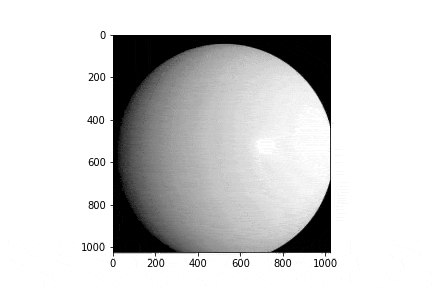
*Laplacian smoothing objective for sphere.obj: 0.00400096819083205*

Значения регуляризатора *Smoothing loss*:

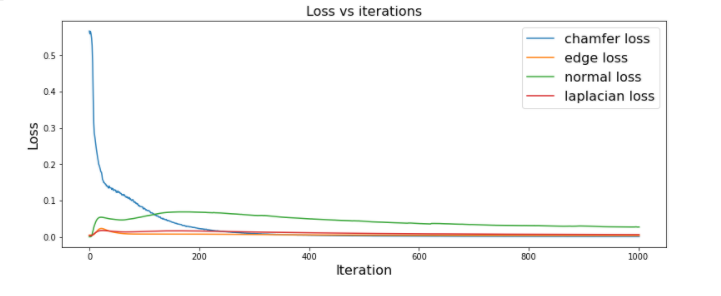
*Smooth regularizer for mickey.obj: 0.2415877878665924*

*Smooth regularizer for sphere.obj: 0.0009780693799257278*

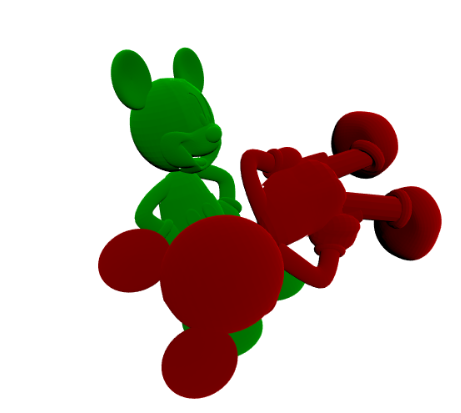
Деформация сферы в модель *mickey.obj.* Для создания анимации используется библиотека celluloid.

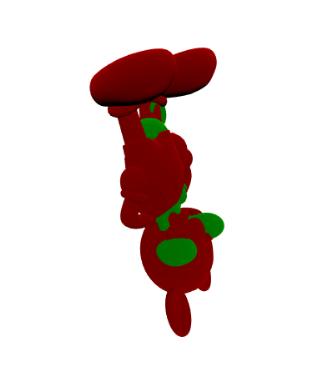


Изменение функций потерь и регуляризаторов в процессе обучения:



# Домашнее задание №3





* Применение метода главных осей инерции (principal axes of inertia):

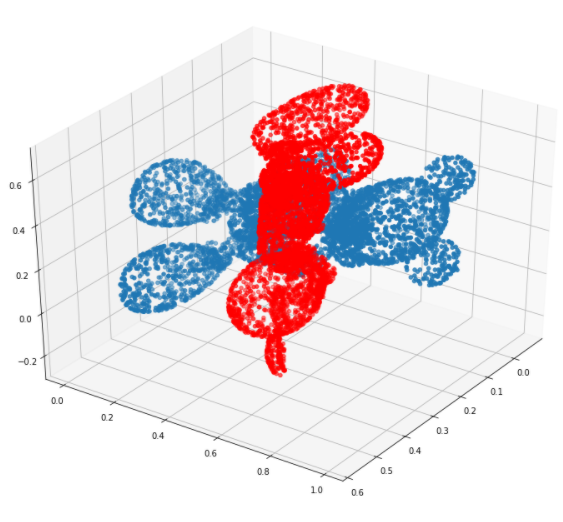
Величина ошибки регистрации: 1.015519187182408e-07

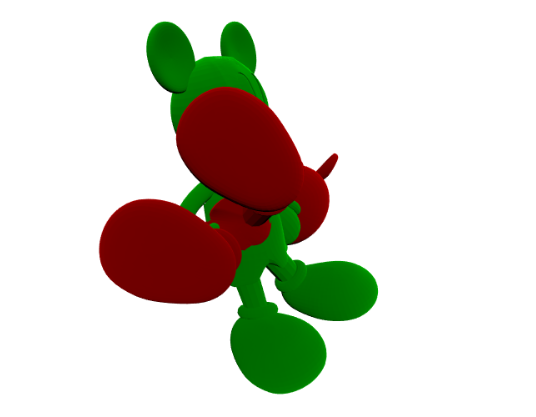
Матрица преобразования модели:

[[-0.45075615 -1.0819739 -1.11683154 1.36914198]

[ 1.43362516 0.161193 -0.73477687 -0.0503725 ]

[-0.60224506 1.19352864 -0.9132095 0.49497544]

[ 0. 0. 0. 1. ]]



# Домашнее задание №4

Восстановление модели по фотографии.

Создал модель своей фотографии используя PIFuHD Demo.

Результат:



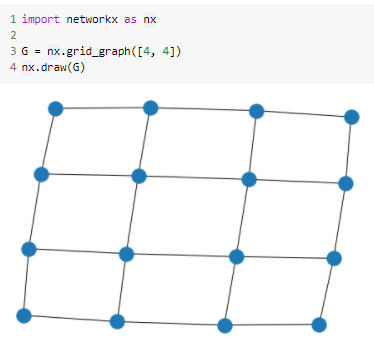
# Домашнее задание №5

Свёртки на графах.

Граф – не Евклидова структура.

Представление информации в нодах или субграфах в виде векторов - проблема. Оказывается, как правило, данные содержатся в многомерном неевклидовом пространстве. А нам нужны векторы в пространстве, в котором мы можем классифицировать, кластеризировать, предсказывать какую-то вероятность и делать кучу других манипуляций ML. Соответственно задача - **перегнать многомерный вектор признаков в низкоразмерный, размерностью d, желательно из R**.

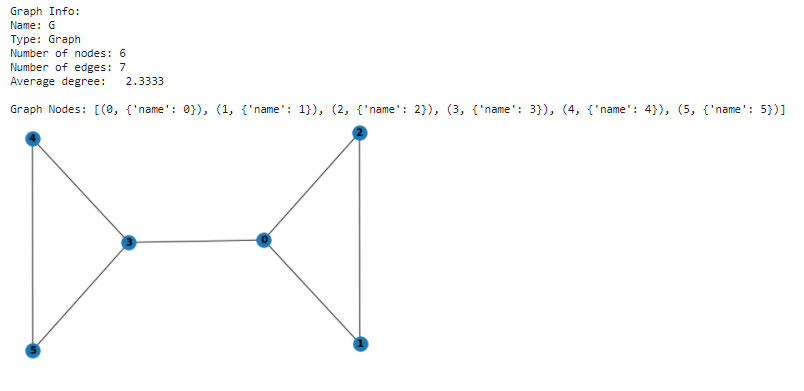
Создание 2D графа 4x4:



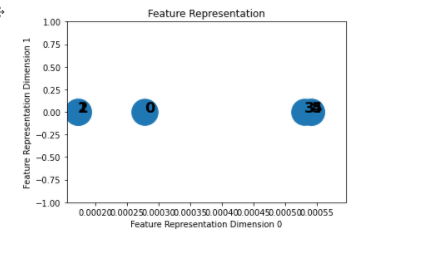
Далее рассмотрим несколько примеров

Пример 1

Исходные данные:

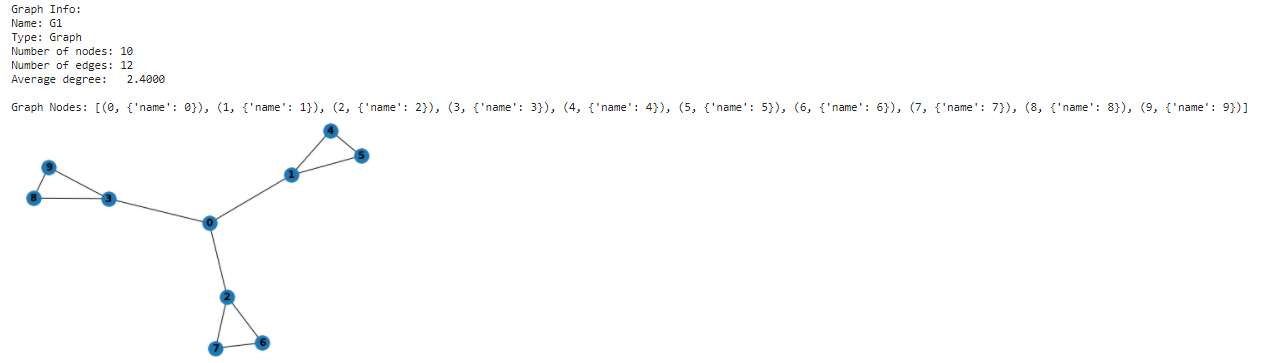


Результат:

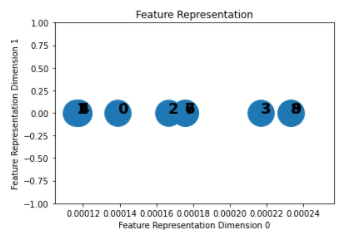


Пример 2

Исходные данные



Результат:

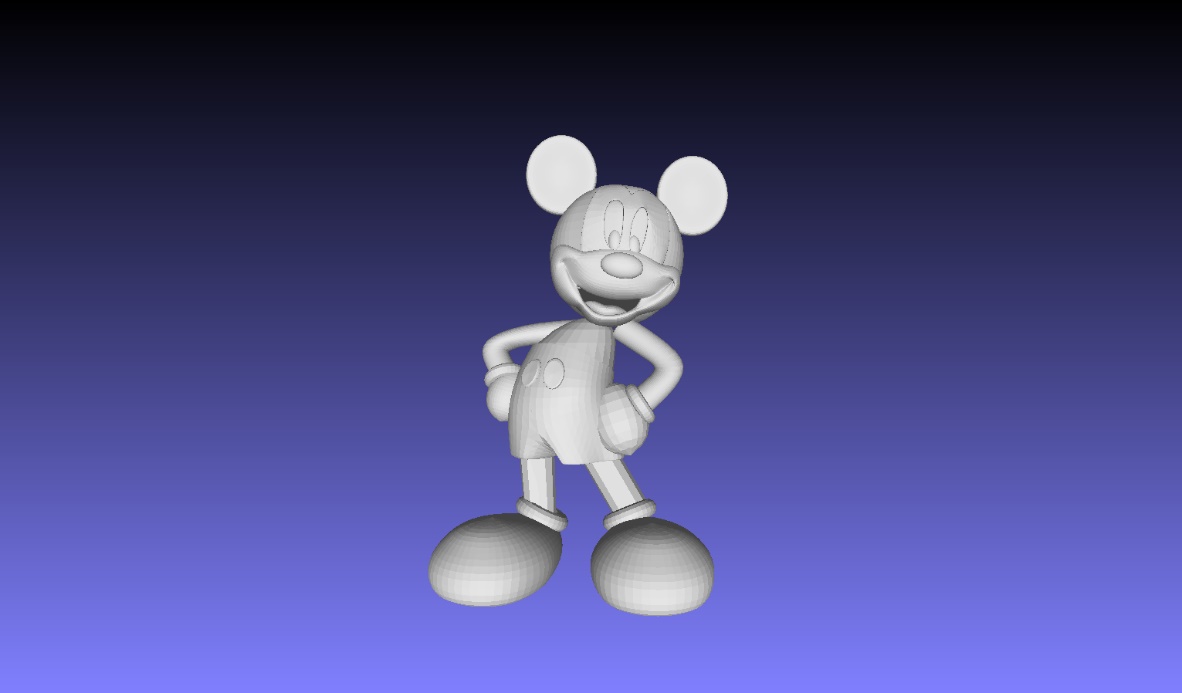


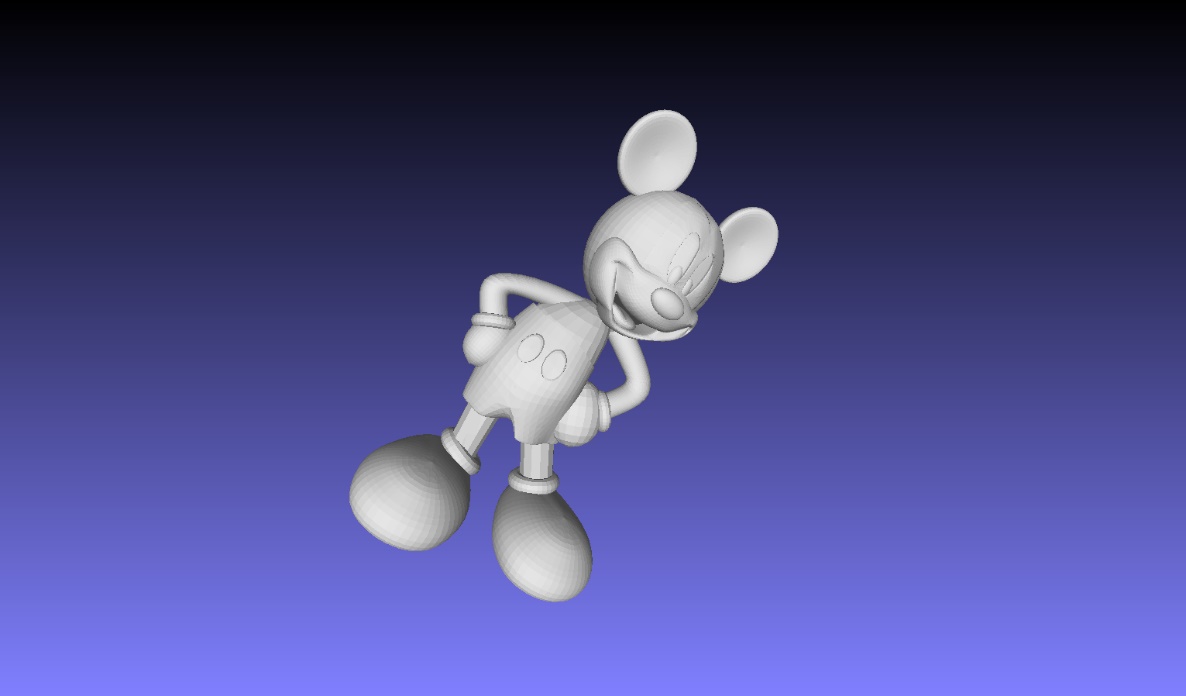
# Домашнее задание №6

Построение карт глубин (MeshLab) с применением быстрого преобразования Фурье (OpenCV):

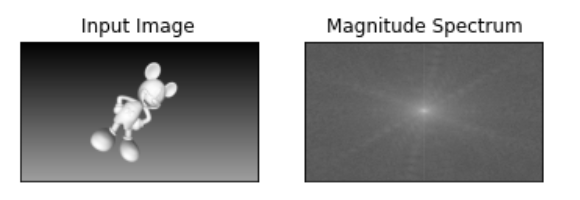
Карты глубины (depth maps) — это снимки, которые фиксируют пространственную информацию: их пиксели содержат в себе расстояния от камеры до соответствующих точек пространства.

Рендеринг в MeshLab:

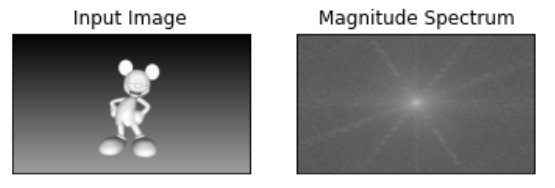




Пример 1



Пример 2



* Источники:
  + MeshLab: <https://www.meshlab.net/>
  + Быстрое преобразование Фурье в OpenCV: <https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_transforms/py_fourier_transform/py_fourier_transform.html>

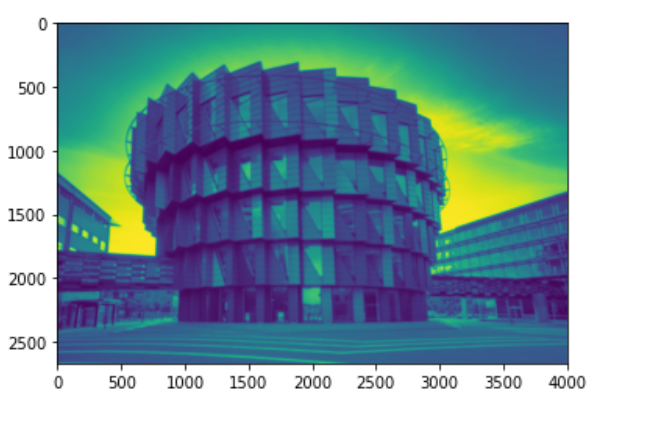
**Домашнее задание №7**

Пример 1

Исходное изображение



Гауссово размытие



Ω = 2



Ω = 5

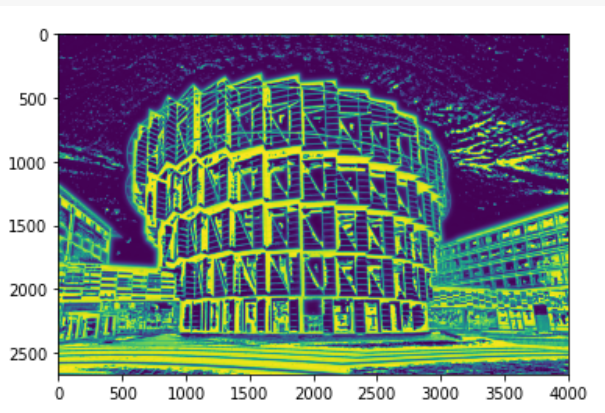


Ω = 10



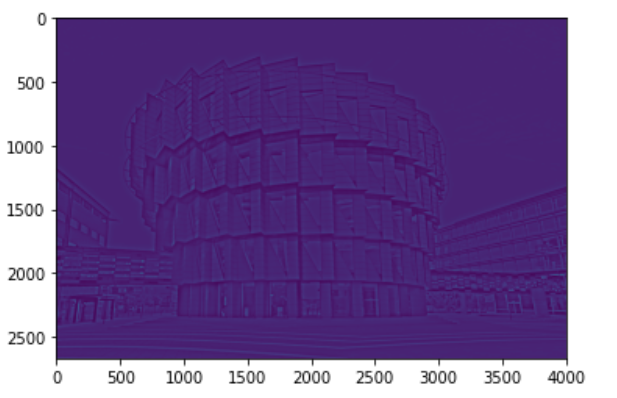
Пример 2

Реализация операции нерезкого маскирования

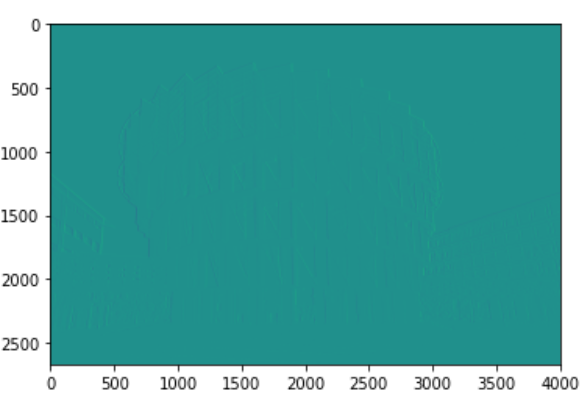


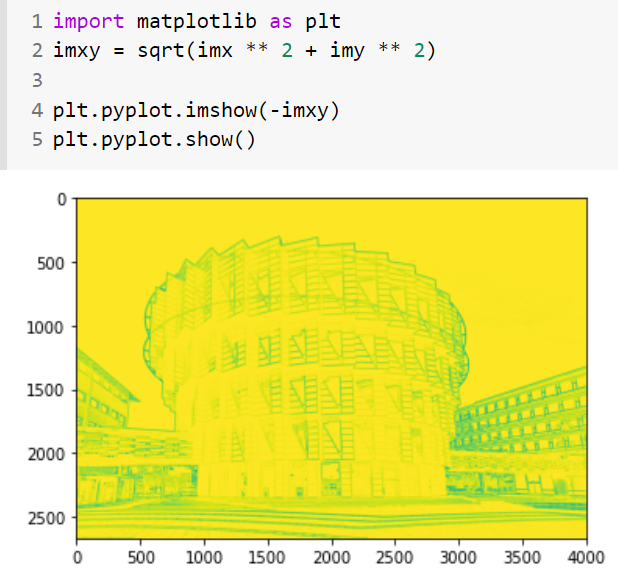
Пример 3

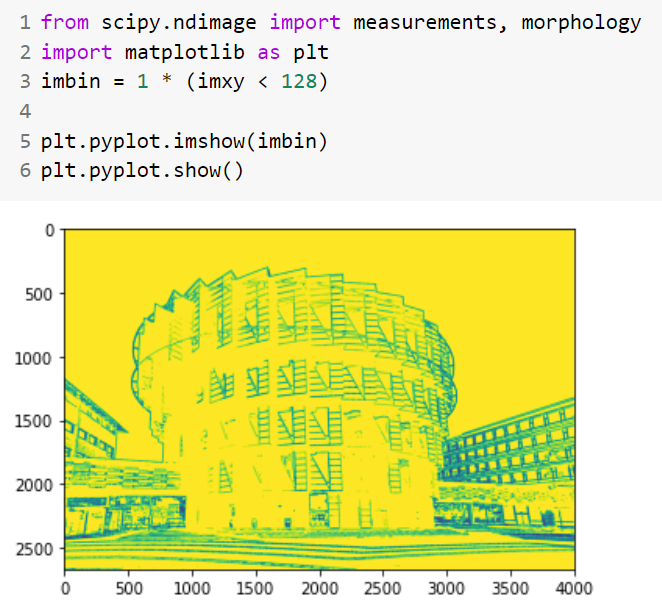
Вычисление изображения - частного



Пример 6

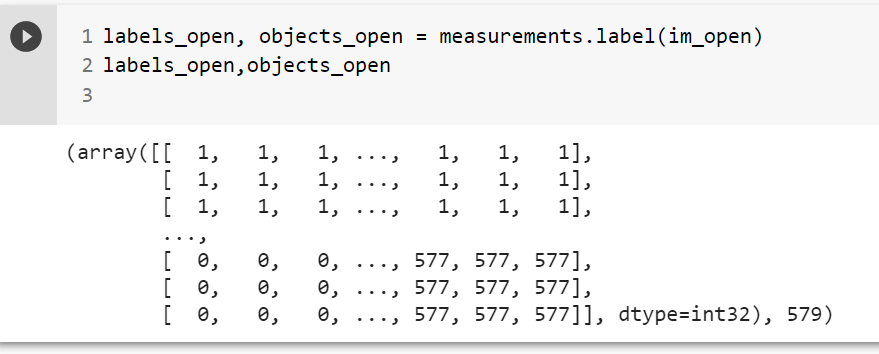


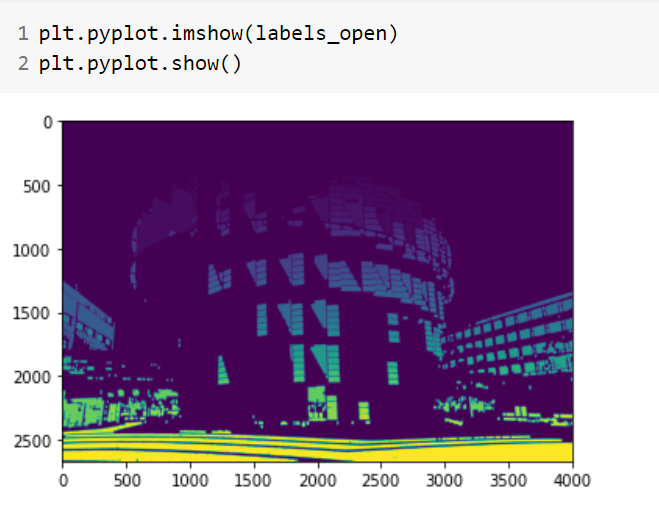


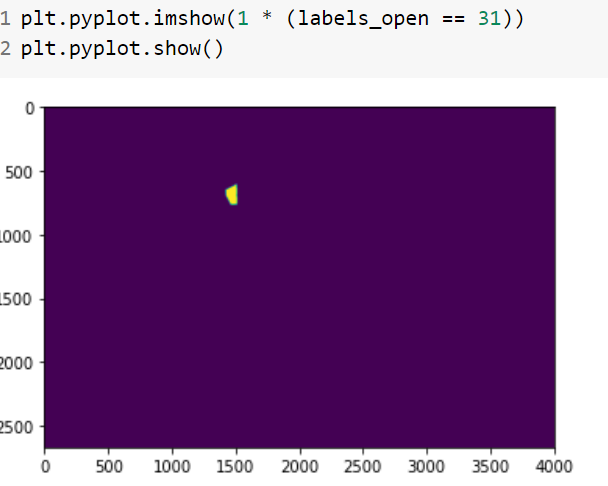


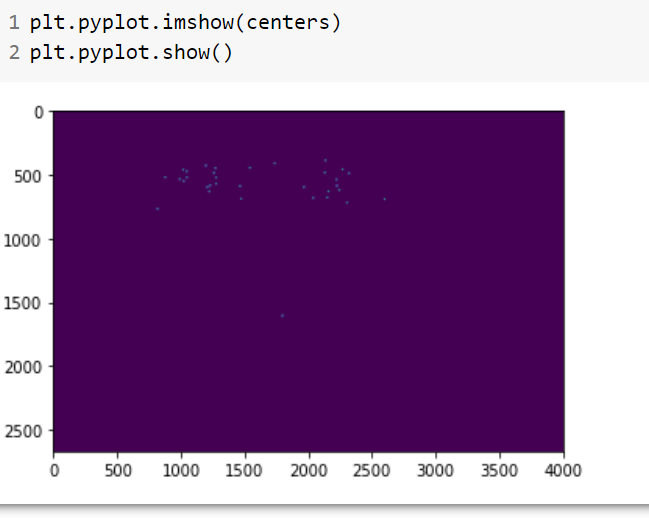


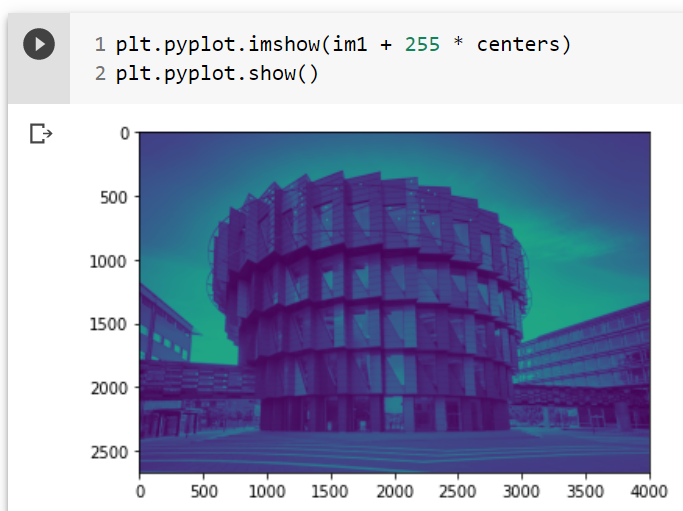
Применение функции label к бинаризованному изображению:

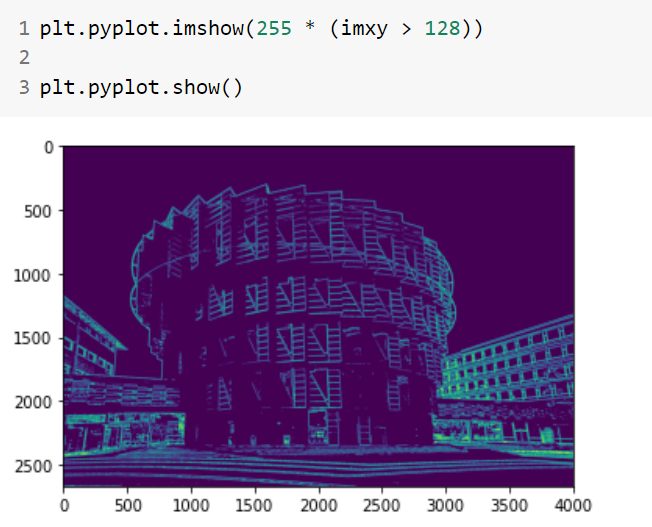


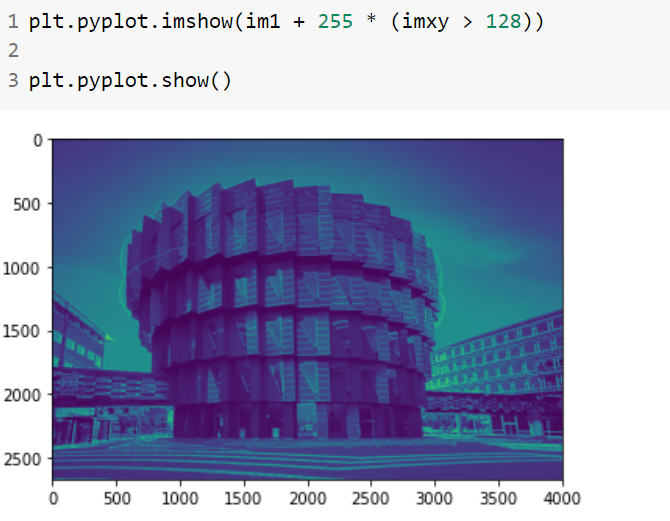












# Приложения

Местоположение кода на [https://github.com/](https://github.com/greyss-mai/Department806)