3D Scanning & Motion Capture

Exercise - 2

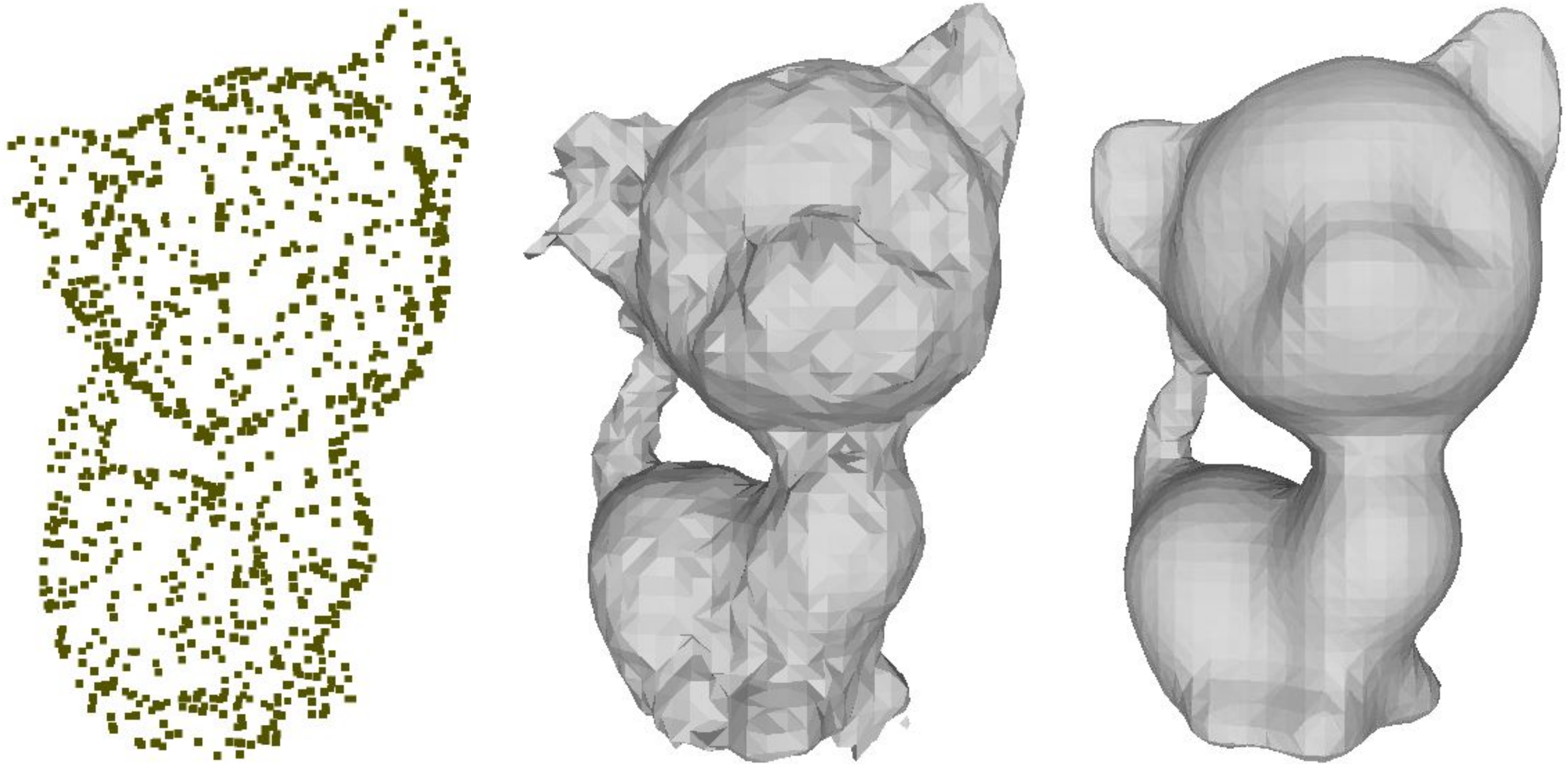
Andrei Burov, Yuchen Rao

Exercises – Overview

1. Exercise $\longrightarrow$ Camera Intrinsics, Back-projection, Meshes
2. **Exercise $\longrightarrow$ Surface Representations**
3. Exercise $\longrightarrow$ Coarse Alignment (Procrustes)
4. Exercise $\longrightarrow$ Optimization
5. Exercise $\longrightarrow$ Object Alignment, ICP

Exercises – Overview (2/5)

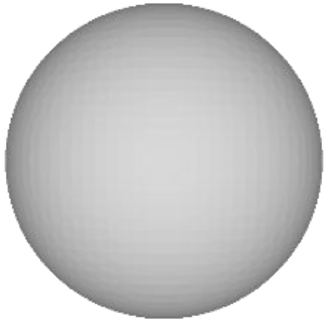
2. Exercise → Surface Representations



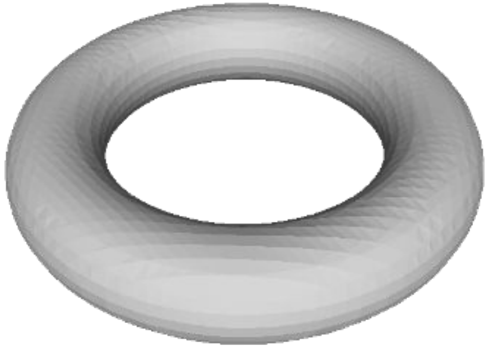
Tasks

1. Project dependencies & CMake configuration
2. Implicit Surfaces
   - Sphere
   - Torus
3. Marching Cubes
   - Improve vertex positions using linear interpolation
4. Hoppe
   - Convert a point cloud to an implicit surface
5. Radial Basis Functions
   - Setup and solve system of linear equations for smoother surfaces

Task 2) Implicit Functions – Sphere / Torus



$$f(x, y, z) = x^2 + y^2 + z^2 - R^2$$

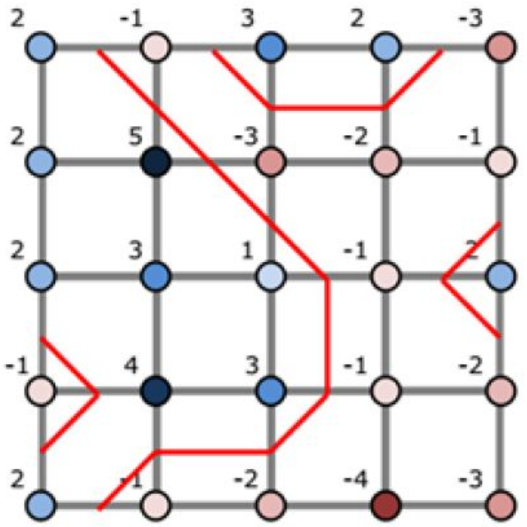


$$f(x, y, z) = (x^2 + y^2 + z^2 + R^2 - a^2)^2 - 4R^2(x^2 + y^2)$$

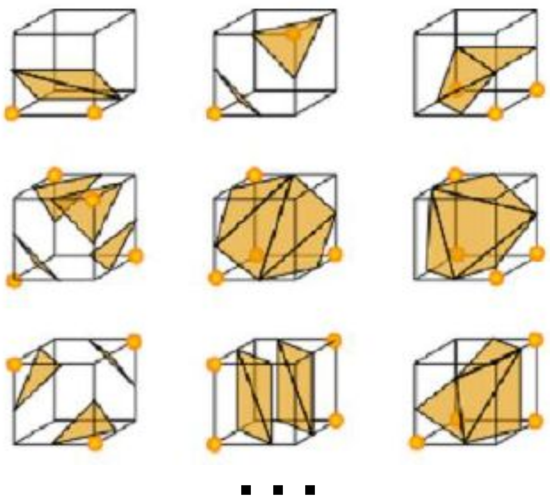
The given equations assume the sphere/torus is centered at the origin.

Task 3) Marching Cubes

- Regular grid/volume → Extract iso-surface
  - Check for zero-crossings within each cell



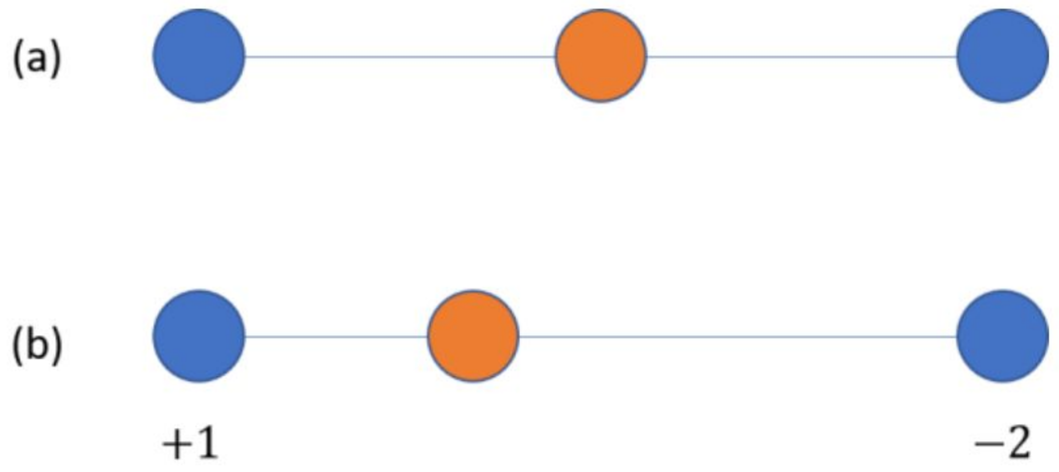
Marching Squares (2D)
16 configurations



Marching Cubes (3D)
256 configurations

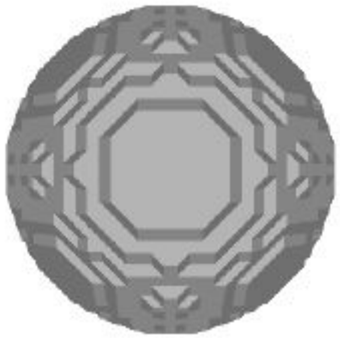
Task 3) Linear Interpolation

- Compute the linear interpolated point using the provided distances
  - (a) shows the basic implementation
  - (b) shows an example with *isolevel* = 0, *valp1* = +1 and *valp2* = −2 .

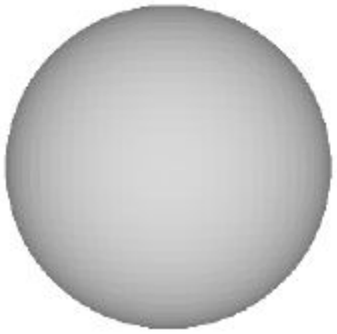


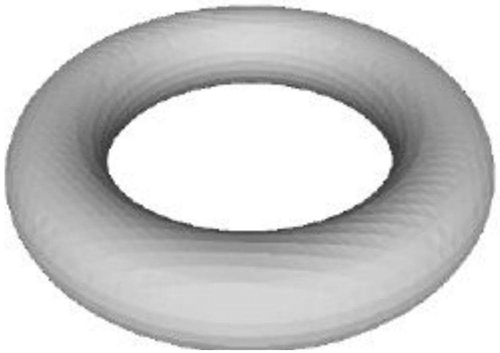
Task 3) Linear Interpolation

- Without linear interpolation
  - i.e. taking midpoint of each edge

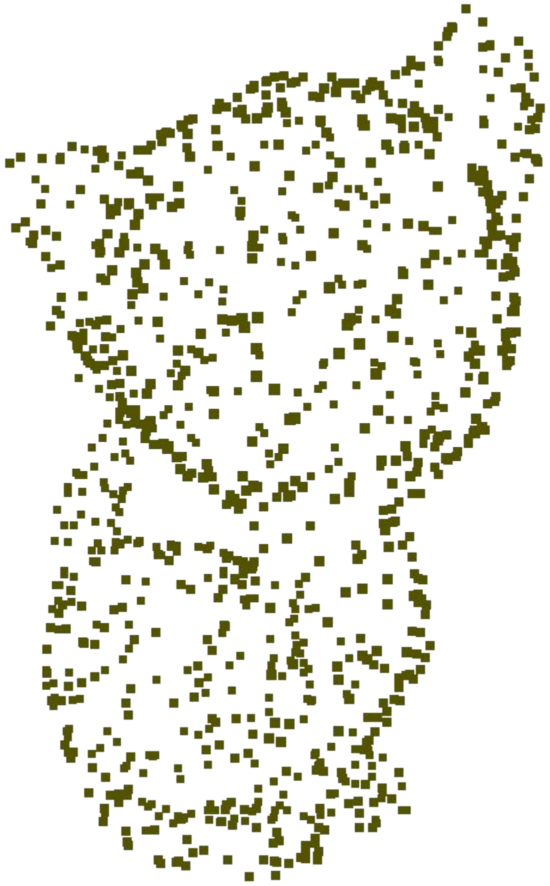


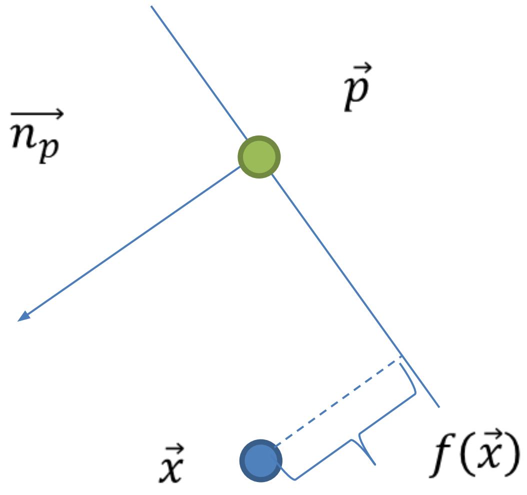
- With linear interpolation





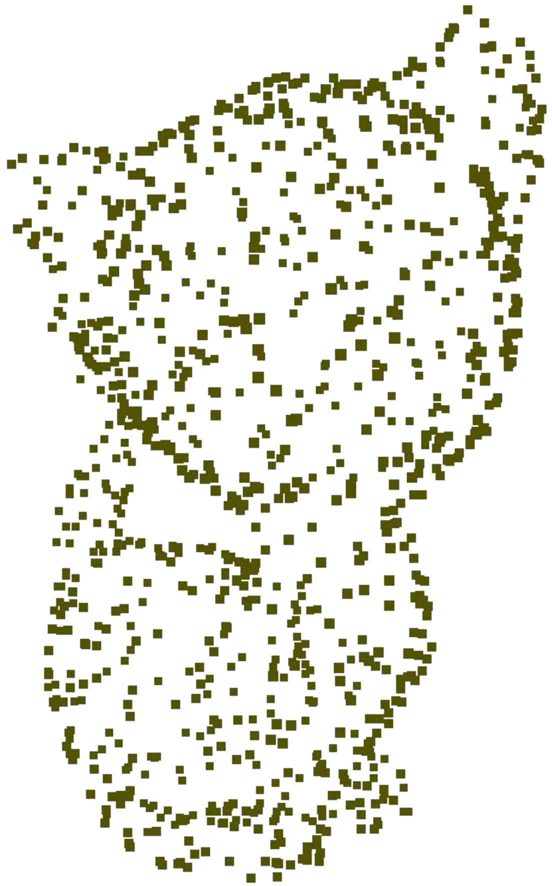
Task 4) Hoppe



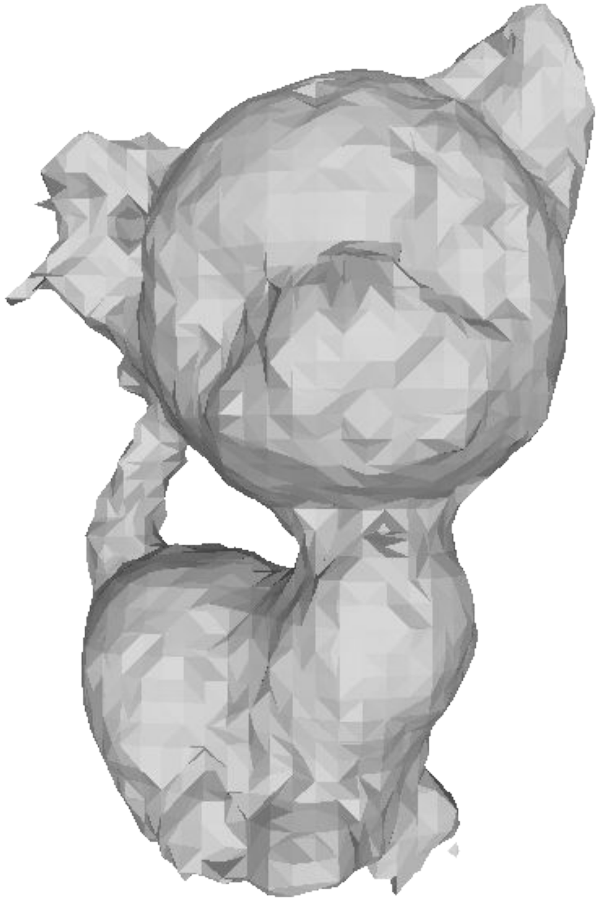


Piecewise linear surface approximation

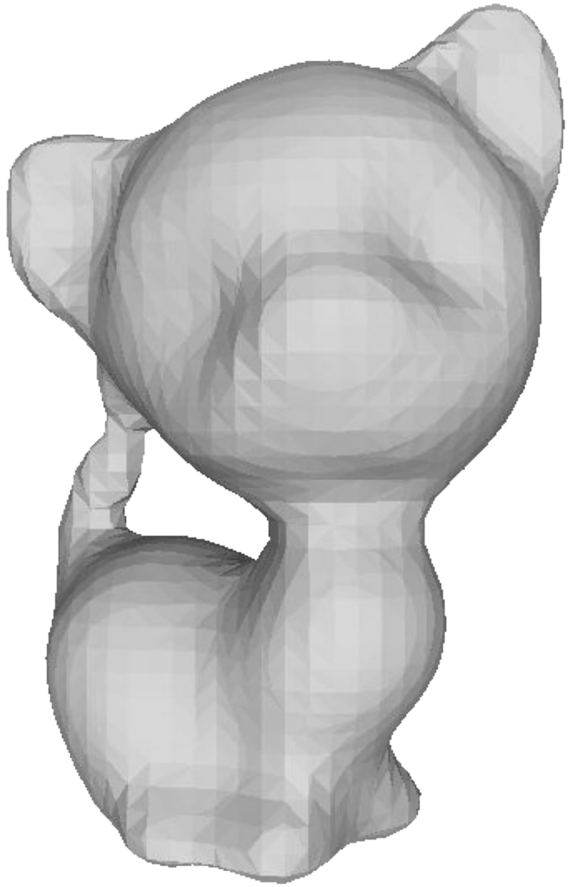
Task 5) Radial Basis Functions (RBF)



Input Points



Hoppe

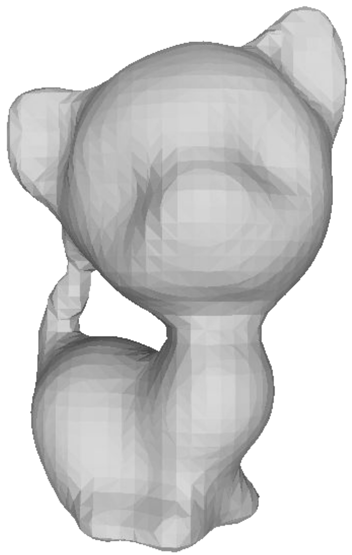


RBF

Task 5) Radial Basis Functions (RBF)

$$f(\vec{x}) = \sum_i \boldsymbol{\alpha_i} \cdot \|\vec{p}_i - \vec{x}\|^3 + \vec{\mathbf{b}} \cdot \vec{x} + \boldsymbol{d}$$

$$\begin{array}{l} \\ \text{on surface} \\ \text{points} \\ \\ \text{off surface} \\ \text{points} \\ \\ \end{array} \underbrace{\begin{bmatrix} \varphi_{1,1} & \cdots & \varphi_{1,\mathrm{n}} & p_{1,x} & p_{1,y} & p_{1,z} & 1 \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \varphi_{\mathrm{n},1} & \cdots & \varphi_{\mathrm{n},\mathrm{n}} & p_{n,x} & p_{n,y} & p_{n,z} & 1 \\ \varphi_{\mathrm{n}+1,1} & \cdots & \varphi_{\mathrm{n}+1,\mathrm{n}} & p_{n+1,x} & p_{n+1,y} & p_{n+1,z} & 1 \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \varphi_{2\cdot\mathrm{n},1} & \cdots & \varphi_{2\cdot\mathrm{n},\mathrm{n}} & p_{2\cdot n,x} & p_{2\cdot n,y} & p_{2\cdot n,z} & 1 \end{bmatrix}}_{A} \cdot \underbrace{\begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_n \\ b_1 \\ b_2 \\ b_3 \\ d \end{bmatrix}}_{\vec{c}} = \underbrace{\begin{bmatrix} h_1 \\ \vdots \\ h_{2\cdot n} \end{bmatrix}}_{\vec{b}}$$



See you next time!