References (Meshes)

- Mario Botsch et al. *Polygon Mesh Processing*. A K Peters, 2010. ISBN: 978-1-56881-426-1. URL: http://www.crcpress.com/product/isbn/9781568814261
  - Chapter 1 (Surface representations)
  - Chapter 2 (Mesh data structures)
  - Chapter 3 (Differential geometry) (overall read)
  - Chapter 5 (Parametrization)
  - Chapter 6 (Remeshing) (overall read)
  - Chapter 7 (Simplification & Approximation) (overall read)
- John F. Hughes et al. *Computer Graphics: Principles and Practice*. 3rd ed. Upper Saddle River, NJ: Addison-Wesley, 2013. ISBN: 978-0-321-39952-6
  - Chapter 8 (A simple way to describe shape in 2D and 3D)
  - Section 23.3 (Catmull-Clark subdivision surfaces)
  - Chapter 24 (Implicit representations of shape)
  - Chapter 25 (Meshes)
  - Chapter 9 (Functions on Meshes)
  - Section 36.4.1 (The Painter's Algorithm)
  - Sections 15.1, 15.2 and 15.4 (Ray casting)
  - Section 31.10 (Radiosity)
- Jamis Buck. *The Ray Tracer Challenge: A Test-Driven Guide to Your First 3D Renderer (Pragmatic Bookshelf)*. English. Paperback. Pragmatic Bookshelf, Mar. 8, 2019. ISBN: 978-1680502718

  Not directly used, but interesting bibliography that explains concepts of raytracing with a completely hands-on approach
- Luiz Henrique de Figueiredo and Jorge Stolfi. "Adaptive Enumeration of Implicit Surfaces with Affine Arithmetic". In: *Computer Graphics Forum* (1996). ISSN: 1467-8659. DOI: 10.1111/1467-8659.1550287
- Luiz Figueiredo and Jorge Stolfi. "Affine Arithmetic: Concepts and Applications". In: (Jan. 2003) *Overall lecture*

Links

- https://en.wikipedia.org/wiki/Texture_mapping (texture mapping)
- https://en.wikipedia.org/wiki/Painter%27s_algorithm (Painter's algorithm)
- https://www.geeksforgeeks.org/painters-algorithm-in-computer-graphics/ (Painter's Algorithm)
- https://en.wikipedia.org/wiki/Ray_tracing_(graphics) (Ray tracing)

Meshes - Practise problems

Answer each question with the appropriate level of detail. For theoretical questions, clearly explain the concepts and, if possible, support your answers with examples. For practical questions, provide the necessary calculations and justify your answers.

1. **(Surface Representations)** Explain the differences between explicit, implicit, and parametric surface representations. Provide examples of each and discuss their advantages and disadvantages.

2. **(Marching Squares)** Describe the *marching squares* algorithm and its application in generating contours of a scalar function defined on a square grid. Include a graphical example illustrating the process. Notice that the *adaptative sampling* that we used in class is not what is being asked in this question.

3. **(Marching Triangles)** Compare and contrast the *marching squares* and *marching triangles* algorithms. Discuss situations in which each would be more appropriate to use.

4. **(Marching Cubes)** Describe the *marching cubes* algorithm and how it can be used to generate isosurfaces in a 3D scalar field. What are the main challenges and how are they resolved? Notice that the *adaptative sampling* that we used in class is not what is being asked in this question.

5. **(OFF and PLY Formats)** Explain the structure of the OFF and PLY formats for mesh representation. What are the advantages and disadvantages of each of them, compared to the other?

6. **(Mesh Data Structures I)** Describe at least three different data structures that can be used to represent meshes. Include in your description:

   - *half-edge*
   - *winged-edge*
   - some easy and wuick representation that you might have used and is not one of the two previous.

7. **(Mesh Data Structures II)** Consider meshes represented with the data structures described in the previous question. Describe algorithms for many useful operations. Analyze conceptual simplicity and computational complexity of each of them.

   The following list serves as *inspiration* for what the *useful operations* can be. By no means this list should be considered as an extensive list. Feel free to think about other potential operations.

   - Iterate through the neighbors of a vertex: Traverse all vertices connected to a given vertex.
   - Split an edge: Insert a new vertex in the middle of an edge, subdividing adjacent triangles.
   - Merge coincident vertices: Combine vertices at the same spatial position to eliminate duplicates and clean up the mesh.

- Remove a vertex and retriangulate adjacent faces: Remove a vertex and adjust surrounding faces to maintain mesh topology.
- Compute vertex normal: Calculate the direction perpendicular to the surface at a vertex by averaging adjacent face normals.
- Identify and fix non-manifold edges: Detect edges shared by more than two faces and correct them to maintain a valid manifold mesh.
- Refine a face: Subdivide a face into smaller faces.
- Simplify a face: Merge faces or vertices to reduce mesh complexity while preserving overall shape.
- Calculate area of a face: Compute the surface area of a polygonal face for geometrical analysis.
- Measure edge length: Determine the length of an edge.
- Identify isolated vertices: Find vertices not connected to any faces (which may indicate errors or incomplete data).
- Swap face vertices: Change the order of vertices in a face.
- Adjust vertex positions.
- Calculate curvature at a vertex: Determine the rate of change of surface normals at a vertex to assess surface features.
- Traverse face adjacency: Iterate over faces sharing an edge.

8. **(Subdivision Surfaces)** Explain the concept of subdivision surfaces and how they are used to generate smooth surfaces from polygonal meshes. Provide an example of a practical application.

9. **(Catmull-Clark Algorithm)** Describe the Catmull-Clark algorithm for surface subdivision. Explain how new vertices are generated and how faces and edges are updated.

10. **(Texture Mapping via Barycentric Coordinates)** Explain the process of texture mapping on a triangle using barycentric coordinates.

11. **(Mesh Parametrization)** What is mesh parametrization? Describe a common method for parametrizing a mesh.

12. **(Topological Operators and Simplification)** Explain what topological operators are in the context of meshes and how they can be used in mesh simplification. Provide two examples of topological operators.

13. **(Implicit Surfaces)** Define what an implicit surface is and describe how it can be used in the modeling of more complex shapes. What are the advantages and challenges of using implicit surfaces?

14. **(Interval Arithmetic)** Explain the concept of interval arithmetic and its potential application in marching squares and marching cubes.

15. **(Ray Tracing)** Explain the ray tracing method for image generation. Mention advantages of this method when compared to the Painter's algorithm.

16. **(Radiosity)** Describe the process of rendering using the radiosity algorithm. Explain why we way that the radiosity method is viewpoint-independent.

17. **(Practical Question: Texture Mapping)** You are given a triangle with vertices at texture coordinates (0,0), (1,0), and (0,1). How would you assign texture coordinates to a point P within the triangle using barycentric coordinates? Provide a numerical example.

18. **(Practical Question: Ray Tracing)** Develop an algorithm for calculating the intersection between a ray and a triangle. Context: the algorithm will be applied in a raytracer.

19. **(Interval Arithmetic Application)** Consider the function $f(x) = x^2 - 2x + 1$. Use interval arithmetic to calculate the range of values of $f(x)$ when $x \in [1, 2]$. Explain the steps of the calculation.

References (Computer Vision)

- Adrian Kaehler and Gary Bradski. *Learning OpenCV 3*. O'Reilly Media, Inc., 2016. ISBN: 9781491937990
  - Chapter 18 (Camera Models and Calibration)
  - Section *The Canny Edge Detector* of Chapter 12 (overall)
  - Chapter 16 (Keypoints and descriptors)

Links

- Camera model and calibration:
  - `https://www.mathworks.com/help/vision/ug/camera-calibration.html`
  - `https://docs.opencv.org/4.x/dc/dbb/tutorial_py_calibration.html`
- Image stitching: `https://www.opencvhelp.org/tutorials/advanced/image-stitching/`
- Marker-based AR:
  - `https://digitalpromise.org/initiative/360-story-lab/360-production-guide/investigate/augmented-reality/getting-started-with-ar/types-of-ar/`
  - `https://medium.com/@sakshi.dumbre31/marker-based-augmented-model-6e1fe1b3759`
- Augmented reality: `https://blog.siggraph.org/tag/augmented-reality/`
- Border detection
  - `https://www.geeksforgeeks.org/image-edge-detection-operators-in-digital-imag`
  - `https://en.wikipedia.org/wiki/Edge_detection`
  - `https://www.mathworks.com/discovery/edge-detection.html`
  - `https://en.wikipedia.org/wiki/Canny_edge_detector`
  - `https://learnopencv.com/edge-detection-using-opencv/`
  - `https://www.geeksforgeeks.org/feature-detection-and-matching-with-opencv-pyt`

Computer Vision - Practise problems

1. **Pinhole Camera Model** Design a pinhole camera system to capture a clear image of an object located 3 meters away. The camera has a 35 mm wide sensor and we want the object to occupy 70% of the sensor's width. What should be the focal length of the pinhole camera?
2. **(Stereo Cameras)** You have a stereo camera system with a baseline (distance between the two cameras) of 10 cm. The focal length of each camera is 50 mm. The pixel coordinates of a point in the left image are $(x_L, y_L) = (150, 200)$, and in the right image, they are $(x_R, y_R) = (130, 200)$. The size of each pixel is 0.01 mm.

   Calculate the 3D coordinates (X, Y, Z) of the point. Provide your calculations and justifications.
3. **(Extracting and Matching Features)** Explain how the Scale-Invariant Feature Transform (SIFT) algorithm extracts keypoints from an image and describe how these keypoints are matched across different images. Explain also how the Harris Corners are extracted from different images and how can be matched.