CSE 274 (WI23)
Selected Topics in Graphics: Discrete Differential Geometry
Albert Chern
Discrete Differential Geometry

- **Instructor:** Albert Chern
- **TA:** Dylan Rowe
- **Course website:** [https://cseweb.ucsd.edu/~alchern/teaching/cse274_wi23/](https://cseweb.ucsd.edu/~alchern/teaching/cse274_wi23/)
- We use **Piazza** and **Gradescope**
Course information

Discrete Differential Geometry

- **Goal:** Differential geometric notations, their discrete theories, and their implementations. Using Houdini software.
- **Applications:** Geometry processing and simulating physical systems.
- **Grade:** HW0–4 (written and coding)
- **Collaboration:** Final submissions should be individual work, but we encourage you to study the math and solve the problems together!
Geometry in Computational World

- Geometry in computations
- What is geometry?
- What is computing?
- Discrete Differential Geometry
- Structure preserving examples
- Roadmap of the course
Geometry in graphics and engineering

Computer fonts

Simulation domain

Computer aided design

Modeling and Animation

Simulation of physical objects
Digital / discrete geometry

- Digital and discrete geometry mimicking smooth surfaces
Digital / discrete geometry

Computational, discrete version

Discrete Differential Geometry

theory for smooth curves & surfaces
Geometry processing

- View discrete surface as a form of “signal data”
  - Traditional signal data: audio & images
  - Geometric signal
  - Upsampling / downsampling / filtering / aliasing
Geometry processing

- Scan
- Process
- Print
- Physics
- Animation / Render
Tasks of geometry processing

- reconstruction
- remeshing
- shape analysis
- parameterization
- filtering
- compression
General theme

Geometry Processing

Partial Differential Equations (PDEs) on general domains
General theme

Geometry
- differential geometry
- geometric analysis
- algebraic topology

Computing
- numerical computations
- physics simulation
- visual data synthesis
- computer aided designs
- control theory
- machine learning
What is Geometry?

- Geometry in computations
- What is geometry?
- What is computing?
- Discrete Differential Geometry
- Structure preserving examples
- Roadmap of the course
What is geometry?

- Relationship between shapes
- Order of spaces
What is geometry? (18th, 19th century)

Differential geometry

- Engineering design
- General relativity
- Computer graphics

Schwarz 1890

photo courtesy Capanna
What is geometry? (19th century)

Non-Euclidean geometry

- Two points connect into a line
- Center and distance gives a circle
- $90^\circ$ is well-defined
- Two lines meet unless parallel

Lobachevsky, Möbius, Grassmann, Beltrami, Lie, Plücker, Cayley, Klein,...
What is geometry? (late 19th early 20th)

Klein’s Erlangen program 1872:

Ingredients of geometry:
- Collection of objects
- Group of transformations
- Invariant notions

Felix Klein 1872  Emmy Noether 1915  Albert Einstein 1916
What is Geometry?

Ingredients of geometry:

- Collection of objects
- Group of transformations
- Invariant notions
  - straight lines, planes
  - distance
  - circles, spheres
  - angles
  - area
  - parallelism
- Non-invariant notions
  - additions $p_1 + p_2$
  - dot product $p_1 \cdot p_2$

Geometry in Euclidean space
What is Geometry?

Ingredients of geometry:
- Collection of objects
- Group of transformations

\[
\begin{align*}
\mathbf{p} &= \begin{bmatrix} x \\ y \\ z \end{bmatrix} \in \mathbb{R}^3 \\
\hat{\mathbf{p}} &= R\mathbf{p}
\end{align*}
\]

- Invariant notions
  - straight lines, planes
  - distance
  - circles, spheres
  - angles
  - area
  - parallelism
  - additions \( \mathbf{p}_1 + \mathbf{p}_2 \)
  - dot product \( \mathbf{p}_1 \cdot \mathbf{p}_2 \)

- Non-invariant notions

Geometry in Euclidean vector space
What is Geometry?

Ingredients of geometry:
- Collection of objects
- Group of transformations

\[
\begin{align*}
\{ \mathbf{p} &= \begin{bmatrix} x \\ y \\ z \end{bmatrix} \in \mathbb{R}^3 \} \\
\tilde{\mathbf{p}} &= A \mathbf{p} \\
\text{invertible matrix}
\end{align*}
\]

- Invariant notions
  - straight lines, planes
  - parallelism
  - additions \( \mathbf{p}_1 + \mathbf{p}_2 \)

- Non-invariant notions
  - distance
  - circles, spheres
  - angles
  - area
  - dot product \( \mathbf{p}_1 \cdot \mathbf{p}_2 \)

Geometry in vector space
What is Geometry?

Ingredients of geometry:
- Collection of objects
- Group of transformations

\[ \begin{align*}
\{ \mathbf{p} &= \begin{bmatrix} x \\ y \\ z \end{bmatrix} \in \mathbb{R}^3 \} \\
\tilde{\mathbf{p}} &= A\mathbf{p} + \mathbf{b} \quad \text{translation} \\
\end{align*} \]

in invertible matrix

Geometry in affine space

Invariant notions
- straight lines, planes
- parallelism
- midpoint \( \frac{\mathbf{p}_1 + \mathbf{p}_2}{2} \)

Non-invariant notions
- distance
- circles, spheres
- angles
- area
- additions \( \mathbf{p}_1 + \mathbf{p}_2 \)
- dot product \( \mathbf{p}_1 \cdot \mathbf{p}_2 \)
What is Geometry?

Ingredients of geometry:

- Collection of objects
  - oriented circles in $\mathbb{R}^2$

- Group of transformations
  - rotate, translate
  - normal offset (propagate wave)

- Invariant notions
  - straight lines
  - oriented contact
  - lengths of common tangents

- Non-invariant notions
  - intersection
  - point

Laguerre geometry of oriented circles
What is Geometry?

Ingredients of geometry:
- Collection of objects
- Group of transformations

Invariant notions:
- tangents, normals
- curvatures

Non-invariant notions:
- particle speed
- particle acceleration

Differential geometry of curves

Particle moving along a trajectory
Reparametrizations
Modern differential geometry

- Provides a version of calculus that is invariant under re-parametrization

$$\text{grad } f = (df)^\#$$

\[\begin{align*}
\dot{x} &= \frac{z (\sqrt{x^2+y^2}+x\theta) - y \sqrt{x^2+y^2} + z^2 \phi}{\sqrt{x^2+y^2} \sqrt{x^2+y^2+z^2}} \\
\dot{y} &= \frac{y (\sqrt{x^2+y^2}+y\theta) + x \sqrt{x^2+y^2} + z^2 \phi}{\sqrt{x^2+y^2} \sqrt{x^2+y^2+z^2}} \\
\dot{z} &= \frac{z \dot{x} - \sqrt{x^2+y^2} \phi}{\sqrt{x^2+y^2} + z^2}
\end{align*}\]
Modern differential geometry

• “Physical laws” in geometries
  ▶ Do you know that:
    number of summits
    + number of bottoms
    – number of saddles = 1

▶ Total Gaussian curvature is shape independent

\[
\int K(\text{dolphin}) = \int K(\text{cow}) = \int K(\text{sphere}) = 4\pi
\]
Modern differential geometry

- Invariants in dynamical system
So, what is geometry?

- Study of shapes and spaces, as well as symmetries and invariants
- Emphasize on structures independent of artificial coordinates and representations
  - Operate calculus at an intuitive level
  - Collection of “physical laws” in geometric shapes and equations
What is Computing?

- Geometry in computations
- What is geometry?
- What is computing?
- Discrete Differential Geometry
- Structure preserving examples
- Roadmap of the course
What is computing?

Three stages:

• Modeling
  ▶ Turning (real)-world problems into mathematical equations

• Mathematical Analysis
  ▶ Making sense from the equations

• Numerical computation
  ▶ Approximate solutions to the equations efficiently and robustly
Classical problems

Three stages:

- **Modeling**
  - Turning (real)-world problems into mathematical equations

- **Mathematical Analysis**
  - Making sense from the equations

- **Numerical computation**
  - Approximate solutions to the equations efficiently and robustly

  - Usually first establish coordinates
  - Then equation is based on the coordinates
    \[
    \left( \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2} \right) \varphi(x, y, z) = f(x, y, z).
    \]
  - Hard to generalize to general spaces without a natural coordinate

  - Hard to see the algebraic structure behind the equations

  - The only available tool to turn the equation into numerics is approximation theory
Geometric approach to computing

Three stages:

- **Modeling**
  - Turning (real)-world problems into mathematical equations

- **Mathematical Analysis**
  - Making sense from the equations

- **Numerical computation**
  - Approximate solutions to the equations efficiently and robustly

Add **Geometry**

- More elegant equations / no coordinate
- Often boils down to linear algebra
- Discrete differential geometry
Discrete Differential Geometry (DDG)

- Geometry in computations
- What is geometry?
- What is computing?
- Discrete Differential Geometry
- Structure preserving examples
- Roadmap of the course
Discretized v.s. Discrete

modeling & analysis

Invariants, Symmetries

discretization

Approximation theory

Use the smooth theory as blueprint and develop a discrete theory

discrete differential geometry

Structure preserving: There are properties which are exact rather than approximation
Discretized v.s. Discrete

Discretize the theory instead of discretizing the equation

Structure preserving: There are properties which are exact rather than approximation
Digital / discrete geometry

- Treat them as discrete geometries in their own right
Examples of Structure-Preserving Discrete Theory

- Geometry in computations
- What is geometry?
- What is computing?
- Discrete Differential Geometry
- Structure preserving examples
- Roadmap of the course
Discrete Gauss–Bonnet Theorem

\[ \int_{\text{entire surface}} \kappa_1 \kappa_2 \ dA = 2\pi \chi \]

\[ \chi = \begin{array}{l}
2 \text{ for sphere} \\
0 \text{ for torus} \\
-2 \text{ for two-holed torus}
\end{array} \]

\[ \sum_{i: \text{vertices}} \Omega_i = 2\pi \chi \]

\[ \Omega_i := 2\pi - \sum_{ijk} \theta_{ij}^{jk} \]
Discretized Example: pendulum equation.

- Energy conservation
  \[ \frac{1}{2} m \dot{\theta}^2 - mg \cos \theta = \text{const} \]
- Integrable system

\[ m \ddot{\theta} = -mg \sin \theta \]
Example: pendulum equation.

\[ m\ddot{\theta} = -mg \sin \theta \]

- High order differential equation solver (4th order Runge–Kutta method)

Given \((\theta_i, v_i = \dot{\theta}_i)\):

\[
\begin{align*}
\theta_{i+1/2}^* &= \theta_i + \frac{\Delta t}{2} v_i \\
v_{i+1/2}^* &= v_i - \frac{\Delta t}{2} \sin \theta_i \\
\theta_{i+1/2}^{**} &= \theta_i + \frac{\Delta t}{2} v_{i+1/2}^* \\
v_{i+1/2}^{**} &= v_i - \frac{\Delta t}{2} \sin \theta_{i+1/2}^*
\end{align*}
\]

\[
\begin{align*}
\theta_{i+1}^{***} &= \theta_i + \Delta t v_{i+1/2}^{**} \\
v_{i+1}^{***} &= v_i - \Delta t \sin \theta_{i+1/2}^{**}
\end{align*}
\]

Output

\[
\begin{align*}
\theta_{i+1} &= \theta_i + \frac{\Delta t}{6} \left( v_i + 2v_{i+1/2}^* + 2v_{i+1/2}^{**} + v_{i+1}^{***} \right) \\
v_{i+1} &= v_i - \frac{\Delta t}{6} \left( \sin \theta_i + 2 \sin \theta_{i+1/2}^* + 2 \sin \theta_{i+1/2}^{**} + \sin \theta_{i+1}^{***} \right)
\end{align*}
\]
Discretized

Example: pendulum equation.

\[ m\ddot{\theta} = -mg \sin \theta \]

- High order differential equation solver (4th order Runge–Kutta method)

Given \((\theta_i, v_i = \dot{\theta}_i)\)

\[ \theta_{i+1/2}^* = \theta_i + \frac{\Delta t}{2} v_i \quad v_{i+1/2}^* = v_i - \frac{\Delta t}{2} \sin \theta_i \]

\[ \theta_{i+1/2}^{**} = \theta_i + \frac{\Delta t}{2} v_{i+1/2}^* \quad v_{i+1/2}^{**} = v_i - \frac{\Delta t}{2} \sin \theta_{i+1/2}^* \]

\[ \theta_{i+1}^{***} = \theta_i + \Delta t v_{i+1/2}^{**} \quad v_{i+1}^{***} = v_i - \Delta t \sin \theta_{i+1/2}^{**} \]

Output \(\theta_{i+1} = \theta_i + \frac{\Delta t}{6} \left( v_i + 2v_{i+1/2}^* + 2v_{i+1/2}^{**} + v_{i+1}^{***} \right)\)

\[ v_{i+1} = v_i - \frac{\Delta t}{6} \left( \sin \theta_i + 2 \sin \theta_{i+1/2}^* + 2 \sin \theta_{i+1/2}^{**} + \sin \theta_{i+1}^{***} \right) \]
Discretized

Example: pendulum equation.

\[ m\ddot{\theta} = -mg \sin \theta \]

- High order differential equation solver (4th order Runge–Kutta method)

Given \((\theta_i, v_i = \dot{\theta}_i)\)

\[
\begin{align*}
\theta_{i+1/2}^* &= \theta_i + \frac{\Delta t}{2} v_i \\
v_{i+1/2}^* &= v_i - \frac{\Delta t}{2} \sin \theta_i \\
\theta_{i+1/2}^{**} &= \theta_i + \frac{\Delta t}{2} v_{i+1/2}^* \\
v_{i+1/2}^{**} &= v_i - \frac{\Delta t}{2} \sin \theta_{i+1/2}^* \\
\theta_{i+1}^{***} &= \theta_i + \Delta t v_{i+1/2}^{**} \\
v_{i+1}^{***} &= v_i - \Delta t \sin \theta_{i+1/2}^{**} \\
\end{align*}
\]

Output

\[
\begin{align*}
\theta_{i+1} &= \theta_i + \frac{\Delta t}{6} \left( v_i + 2v_{i+1/2}^* + 2v_{i+1/2}^{**} + v_{i+1}^{***} \right) \\
v_{i+1} &= v_i - \frac{\Delta t}{6} \left( \sin \theta_i + 2 \sin \theta_{i+1/2}^* + 2 \sin \theta_{i+1/2}^{**} + \sin \theta_{i+1}^{***} \right) \\
\end{align*}
\]

\(\Delta t = 0.9\)
Discrete

Example: pendulum equation.

\[ m\ddot{\theta} = -mg \sin \theta \]

- A 2nd order discretization

\[
\frac{\theta_{i-1} - 2\theta_i + \theta_{i+1}}{\Delta t^2} = -\sin \theta_i
\]

\[ \Delta t = 0.1 \]
Discrete

Example: pendulum equation.

\[ m\ddot{\theta} = -mg \sin \theta \]

- A 2nd order discretization

\[ \frac{\theta_{i-1} - 2\theta_i + \theta_{i+1}}{\Delta t^2} = -\sin \theta_i \]

✓ Energy conservation in “asteroid belts”
Discrete

Example: pendulum equation.
\[ m\ddot{\theta} = -mg \sin \theta \]

- A 2nd order discretization

\[ \frac{\theta_{i-1} - 2\theta_i + \theta_{i+1}}{\Delta t^2} = -\sin \theta_i \]

Least action principle
\[ \int \left( \frac{m}{2} \dot{\theta}^2 + mg \cos \theta \right) dt \]

Energy conservation in “asteroid belts”

First introduce discrete action, then derive the least action paths
Discrete

Example: pendulum equation.

\[ m\ddot{\theta} = -mg \sin \theta \]

- Another 2nd order discretization

\[
\frac{\theta_{i-1} - 2\theta_i + \theta_{i+1}}{\Delta t^2} = -\sin \theta_i
\]

\[
\frac{\theta_{i-1} - 2\theta_i + \theta_{i+1}}{\Delta t^2} = 4 \arg \left( 1 + \frac{\Delta t^2}{4} e^{-i\theta_i} \right)
\]

\[ \checkmark \text{Integrable system} \]

\[ \Delta t = 0.1 \]
Example: pendulum equation.
\[ m\ddot{\theta} = -mg \sin \theta \]

- Another 2nd order discretization

\[
\frac{\theta_{i-1} - 2\theta_i + \theta_{i+1}}{\Delta t^2} = -\sin \theta_i
\]

\[
\frac{\theta_{i-1} - 2\theta_i + \theta_{i+1}}{\Delta t^2} = 4 \arg \left( 1 + \frac{\Delta t^2}{4} e^{-i\theta_i} \right)
\]

✓ Integrable system
Discretized v.s. Discrete

**general**
- 4th order Runge–Kutta
- Variational integrator
- Discrete integrable system

**no-structure**
- quantitative
- high-precision

**Δt = 0.1**

**Δt = 0.9**

**rare**
- structure preserving
- qualitative
- exact
Discretely perfect absorbing material

“A reflectionless discrete perfectly matched layer”
Journal of Computational Physics, 2019
Chern 2019
A reflectionless discrete perfectly matched layer
*J. Comp. Phys.*
Discrete complex analysis

Chern 2019
A reflectionless discrete perfectly matched layer
*J. Comp. Phys.*
Roadmap of the Course

- Geometry in computations
- What is geometry?
- What is computing?
- Discrete Differential Geometry
- Structure preserving examples
- Roadmap of the course
Roadmap of the course

• We will use Houdini software for implementation
NCSA Advanced Visualization Lab
https://youtu.be/T_0ICxROM0Q
Homework 0

• Implementation part of the homework: Houdini Software

▶ Watch our tutorial (link in the syllabus on the course website)

▶ HW0 due next Thursday: upload a screenshot/render of some cool mathematical visualization.
Install Houdini & SciPy (numerical linear algebra for sparse matrices)

Houdini FX

Houdini is a 3D animation software by SideFX that is broadly used in filming industry. Its built-in shader language (VEX), geometric data structures, and its integration with Python (where one can call numerical linear algebra library such as SciPy) make it a powerful tool for scientific computing and geometry processing. Moreover Houdini comes with industry standard renderer allowing stunning visualization. Its free apprentice license is fully functional, with only a few limitations in the resolution of rendering preventing commercial uses. Houdini has become a frequently used software for not only research demonstration, but also for teaching and generating illustrations for lectures and talks.

A terrain with a shape of Mandelbrot set created in Houdini

Lecture Note

-  Houdini Introduction Lecture Note
-  Houdini Tech Blog

Getting Started

To get started, first we recommend rummaging around the SideFX website and searching for tutorials on YouTube. We also recommend that you work through some of the tutorials on the Houdini Tech Blog, a course blog for "Visualization in Mathematics" in TU Berlin. For example you can work through all the tutorials in the "Introduction" category. There are more tutorials on that site and if you feel inspired check them out. All come with Houdini files that allows you to load the completed tutorials if you wish.

Our lecture note listed above teaches you how to create simple geometry inside Houdini using the VEX language; basic scene setup with simple lighting and camera as well as manipulating surface appearance for rendering.

The complete documentation is also available on the SideFX website.

A note on installing Houdini (and SciPy)

1. Find the Python directory. Open Houdini. On any dock (subwindow) add a tab "Python Shell." In the Python shell, type
   ```python
   >>> import os
   >>> pathos.path.abspath(os.__file__)
   >>> print(path)
   ```
   The result should be something like
   `/Applications/Houdini/Houdini18.0.499/Frameworks/Python.framework/Versions/Current/Lib/python2.7/os.py`

2. Open Terminal, and go to the bin directory `"/Applications/Houdini/Houdini18.0.499/Frameworks/Python.framework/Versions/Current/bin` (according to the path shown on Step 1.)
   ```bash
   $ cd /Applications/Houdini/Houdini18.0.499/Frameworks/Python.framework/Versions/Current/bin
   ```

3. Use the local python to install the Python Package Installer (pip). In that directory execute (you will need to enter your root password for the superuser do (sudo) command)
   ```bash
   $ sudo curl https://bootstrap.pypa.io/get-pip.py -o get-pip.py
   $ sudo ./python get-pip.py
   ```

4. Use the local pip to install SciPy.
   ```bash
   $ sudo ./pip install scipy
   ```

5. Back to the Houdini Python shell (no need to restart Houdini). Now
   ```python
   >>> import scipy
   ```
   should give no error.
Install SciPy for Houdini (link in syllabus)

- Install Houdini & SciPy (numerical linear algebra for sparse matrices)
Roadmap of the course

- We will use Houdini software for implementation
- **Topic 1** Triangle meshes

\[ |V| - |E| + |F| = \chi \]
Roadmap of the course

- We will use Houdini software for implementation
- **Topic 1** Triangle meshes
- **Topic 2** Discrete exterior calculus, discrete Laplacian

<table>
<thead>
<tr>
<th></th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-4</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>

Laplace filter on images

$\mathbf{d}^T \star \mathbf{d}$
Roadmap of the course

• We will use Houdini software for implementation

• **Topic 1** Triangle meshes

• **Topic 2** Discrete exterior calculus, discrete Laplacian

• **Topic 3** Continuous exterior calculus, PDEs on manifolds

\[
\frac{1}{r^2} \frac{\partial}{\partial r} \left( r^2 \frac{\partial f}{\partial r} \right) + \frac{1}{r^2 \sin \theta} \frac{\partial}{\partial \theta} \left( \sin \theta \frac{\partial f}{\partial \theta} \right) + \frac{1}{r^2 \sin^2 \theta} \frac{\partial^2 f}{\partial \phi^2}
\]

\[d \star df\]
Roadmap of the course

- We will use Houdini software for implementation
- **Topic 1** Triangle meshes
- **Topic 2** Discrete exterior calculus, discrete Laplacian
- **Topic 3** Continuous exterior calculus, PDEs on manifolds
- **Topic 4** Differential geometry of curves and surfaces
Roadmap of the course

• We will use Houdini software for implementation
• **Topic 5** Linear algebraic topology
Roadmap of the course

- We will use Houdini software for implementation
- **Topic 5** Linear algebraic topology
- **Topic 6** Vector field design, gauge theory