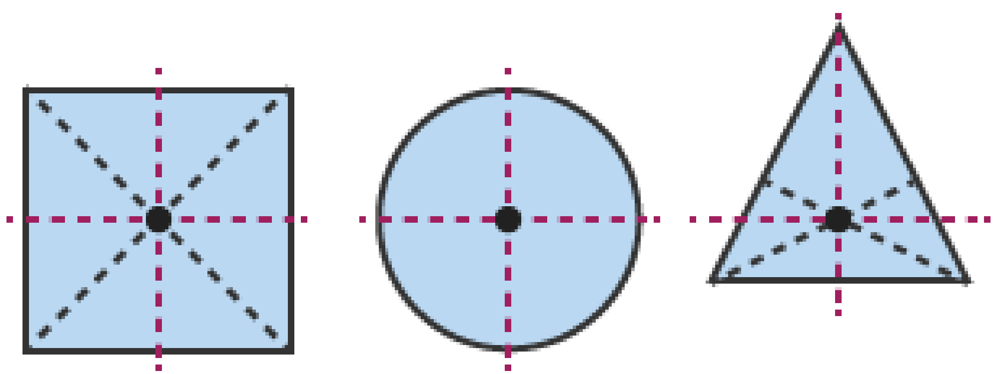
Barycentric Coordinates and Parameterization

Center of Mass

"Geometric center" of object

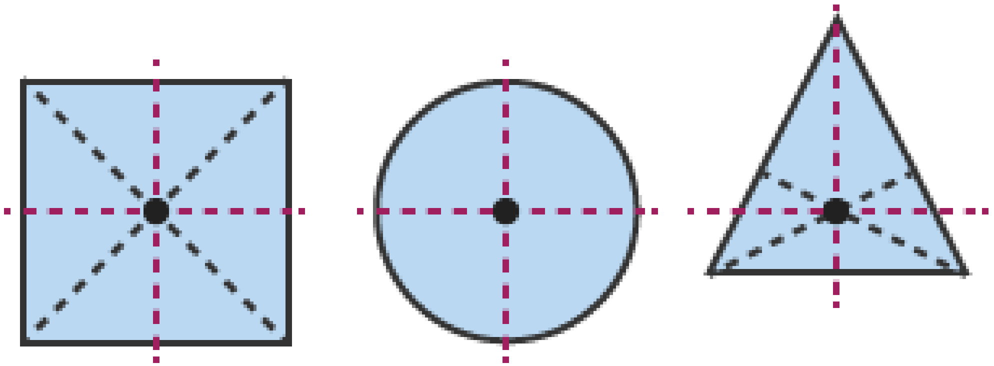


Center of Mass

“Geometric center” of object

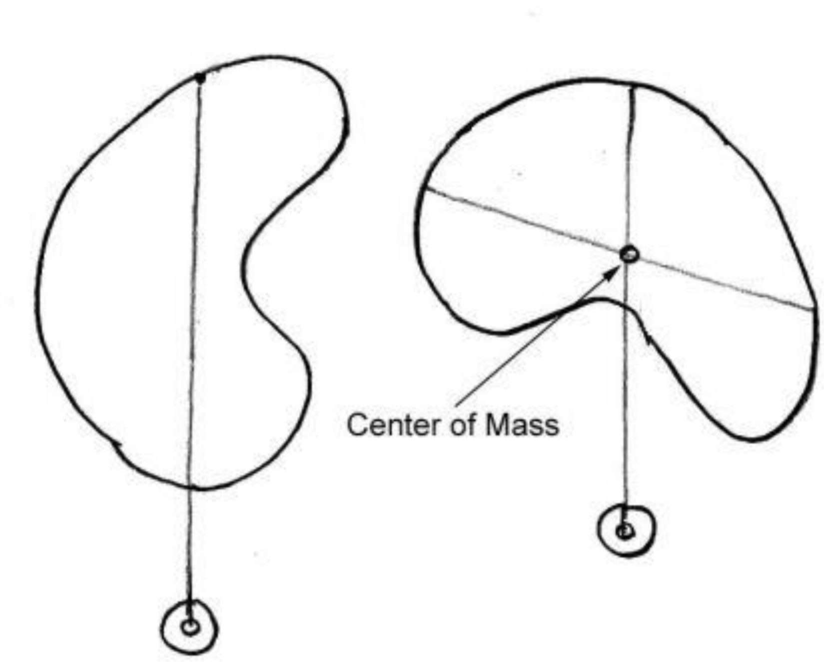
Object can be balanced on CoM

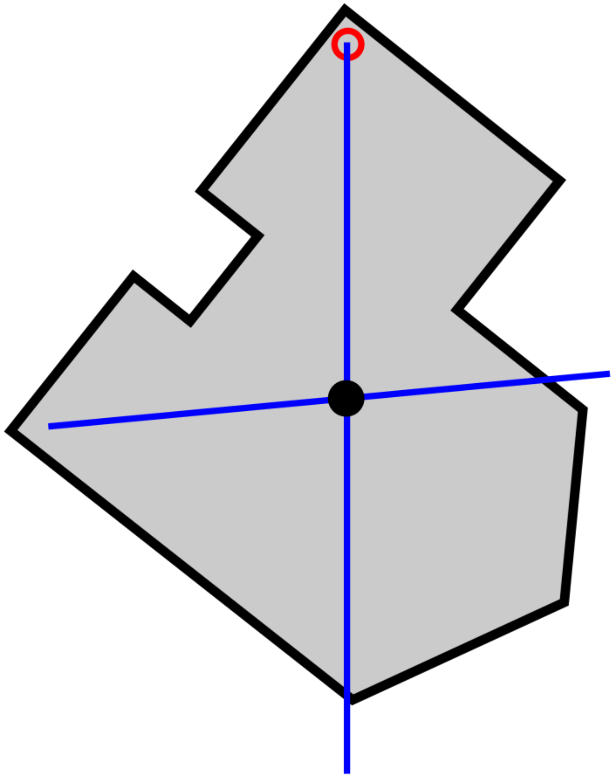
How to calculate?



Finding the Center of Mass

Plumb line method

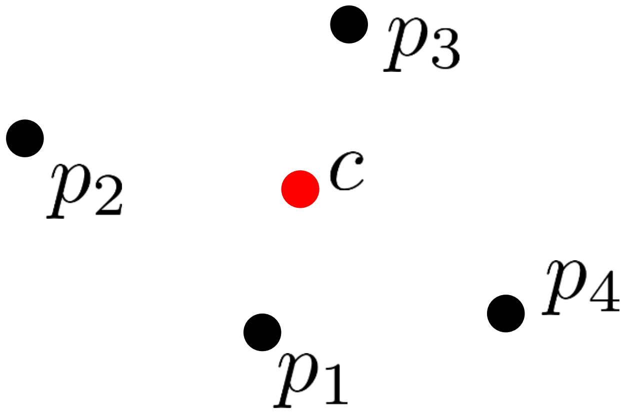




Special Case: Points

CoM is average

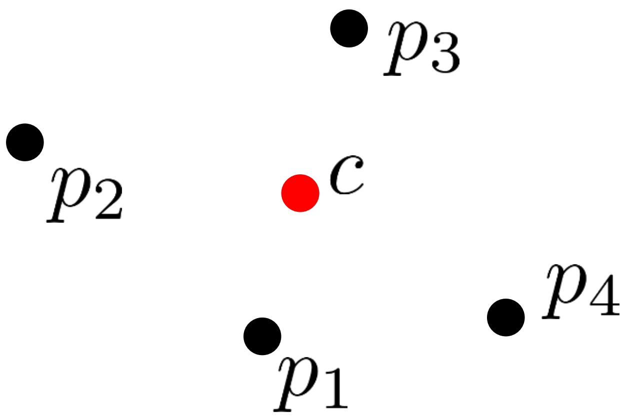
$$c = \frac{1}{n}\sum_{i=1}^{n} p_i$$



Special Case: Points

CoM is average

$$c = \frac{1}{n} \sum_{i=1}^{n} p_i$$

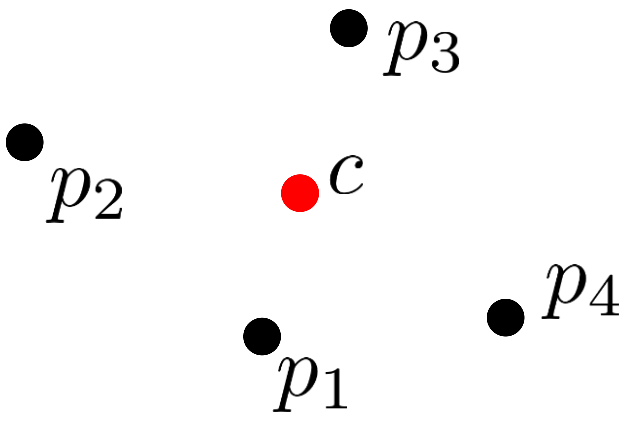


Center of mass is inside **convex hull**

Special Case: Points

CoM is average

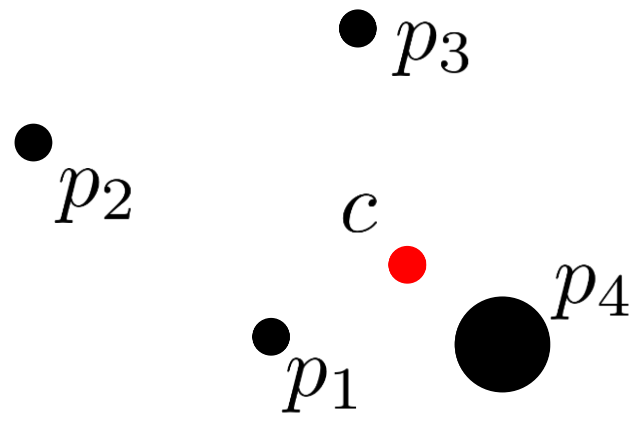
$$c = \frac{1}{n}\sum_{i=1}^{n} p_i$$



Center of mass is inside **convex hull**

What if points have different mass?

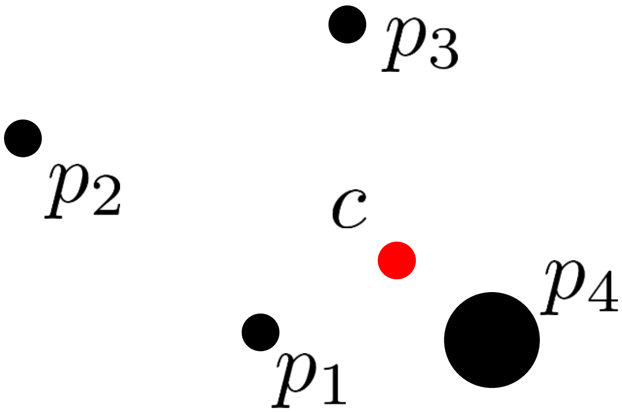
Special Case: Points



Special Case: Points

Weighted average

$$c = \frac{\sum_{i=1}^{n} m_i p_i}{\sum_{i=1}^{n} m_i}$$



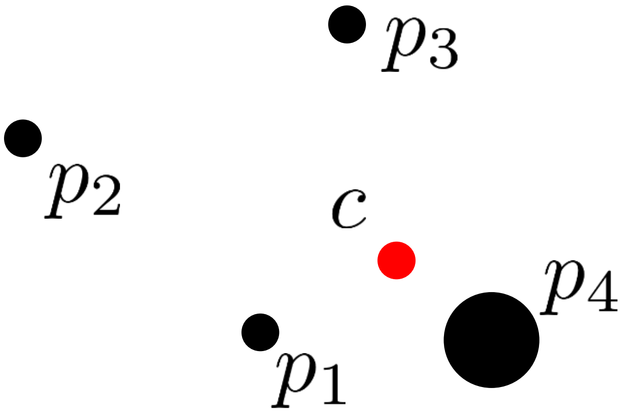
Still in convex hull

Scaling the masses doesn't affect CoM

Special Case: Points

Weighted average

$$c = \frac{\sum_{i=1}^{n} m_i p_i}{\sum_{i=1}^{n} m_i}$$



Still in convex hull

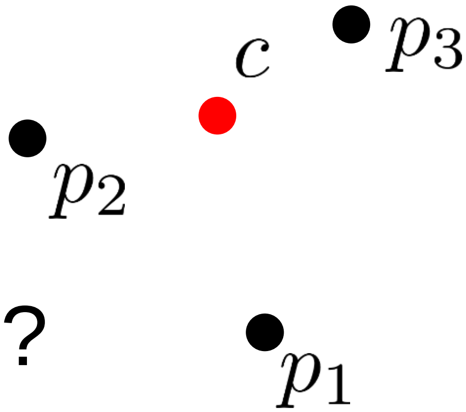
Scaling the masses doesn’t affect CoM

- can assume masses sum to one

Inverse Problem

Given three points p_i and a target point c:

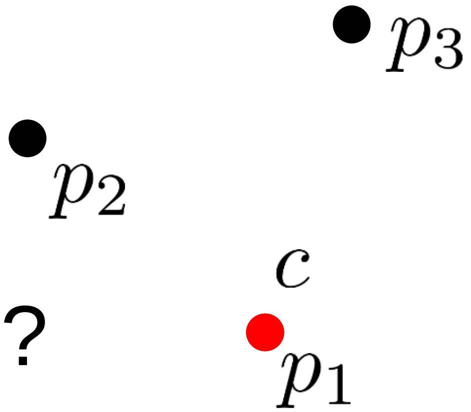
For what masses is c the CoM?



Inverse Problem

Given three points p_i and a target point c:

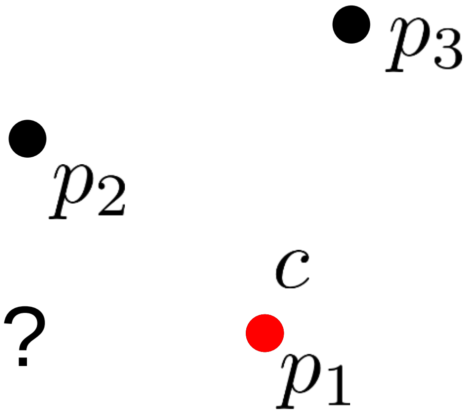
For what masses is c the CoM?



Special case 1: $c = p_i$

Inverse Problem

Given three points p_i and a target point c:

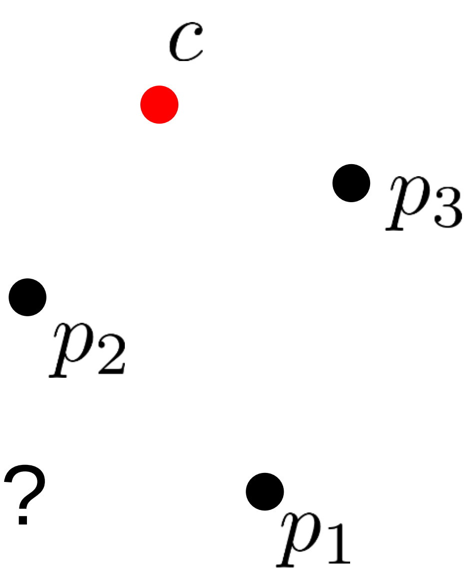


For what masses is c the CoM?

Special case 1: $c = p_i$

$$m_j = \begin{cases} 1, & j = i \\ 0, & j \neq i \end{cases}$$

Inverse Problem



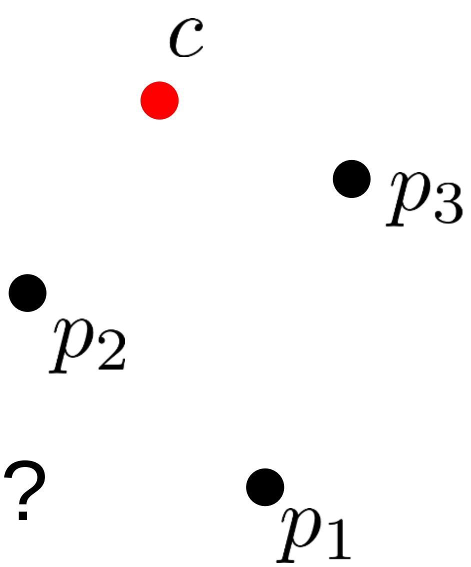
Given three points p_i and a target point c:

For what masses is c the CoM?

Special case 1: $c = p_i$

Special case 2: c outside triangle

Inverse Problem



Given three points p_i and
a target point c:

For what masses is c the CoM?

Special case 1: $c = p_i$

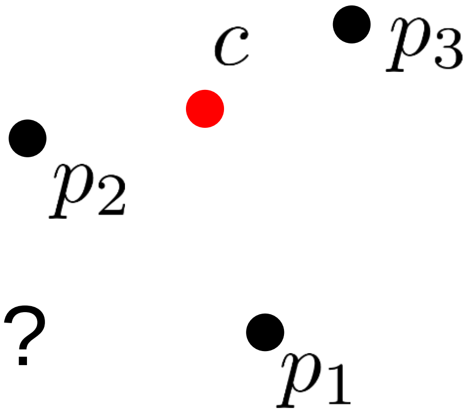
Special case 2: c outside triangle

- not possible (needs antigravity…)

Inverse Problem

Given three points p_i and a target point c:

For what masses is c the CoM?



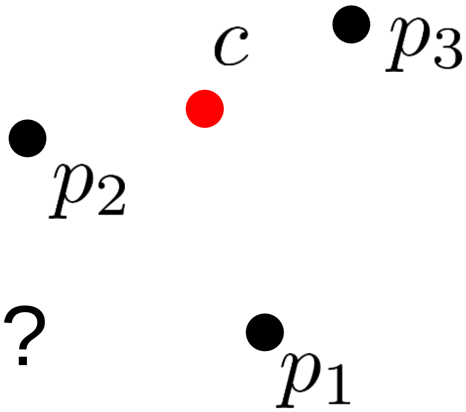
$$c = \sum_i m_i p_i, \quad \sum m_i = 1$$

Observation: $m_1 = 1 - m_2 - m_3$

Inverse Problem

Given three points p_i and a target point c:

For what masses is c the CoM?

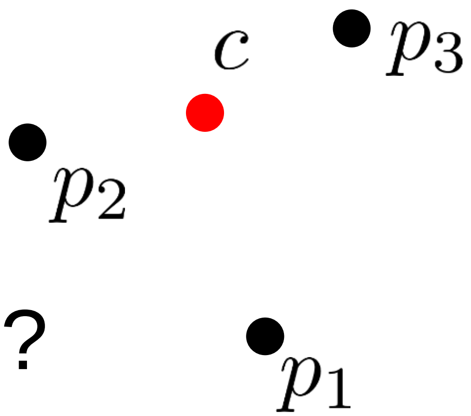


$$c = (1 - m_2 - m_3)p_1 + m_2 p_2 + m_3 p_3$$

Inverse Problem

Given three points p_i and a target point c:

For what masses is c the CoM?



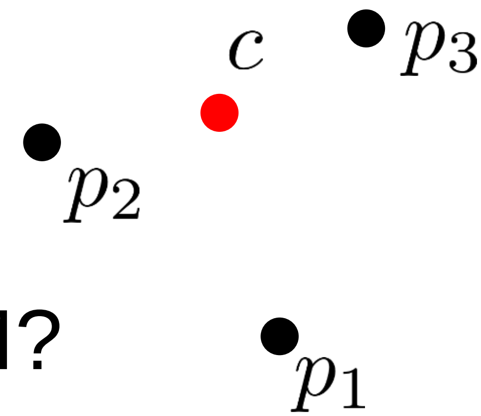
$$c = (1 - m_2 - m_3)p_1 + m_2 p_2 + m_3 p_3$$

$$c - p_1 = m_2(p_2 - p_1) + m_3(p_3 - p_1)$$

Inverse Problem

Given three points p_i and a target point c:

For what masses is c the CoM?



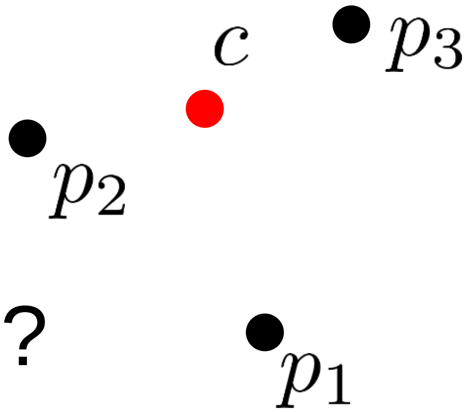
$$c = (1 - m_2 - m_3)p_1 + m_2 p_2 + m_3 p_3$$

$$c - p_1 = m_2(p_2 - p_1) + m_3(p_3 - p_1)$$

$$c - p_1 = \begin{bmatrix} p_2 - p_1 & p_3 - p_1 \end{bmatrix}_{2\times 2} \begin{bmatrix} m_2 \\ m_3 \end{bmatrix}$$

Inverse Problem

Given three points p_i and a target point c:

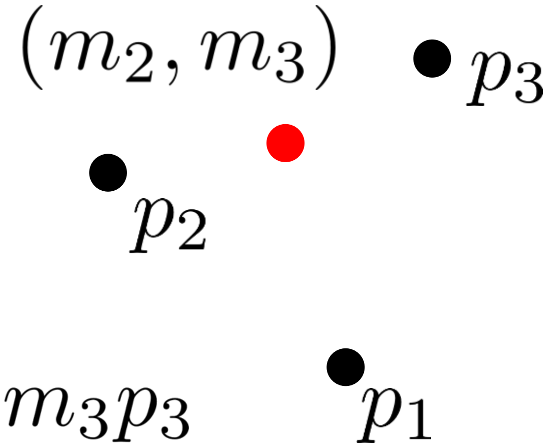


For what masses is c the CoM?

$$\begin{bmatrix} m_2 \\ m_3 \end{bmatrix} = \begin{bmatrix} p_2 - p_1 & p_3 - p_1 \end{bmatrix}^{-1} [c - p_1]$$

These are **barycentric coordinates** of c

Barycentric Coordinates



Can be interpreted as

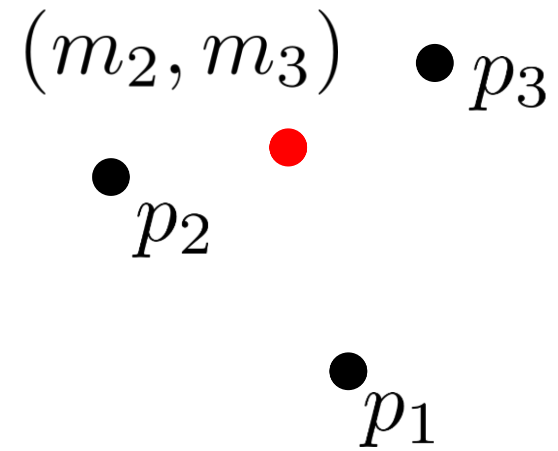
- weighted point sum

$$(1 - m_2 - m_3)p_1 + m_2 p_2 + m_3 p_3$$

- point in **edge coordinates**

$$p_1 + m_2(p_2 - p_1) + m_3(p_3 - p_1)$$

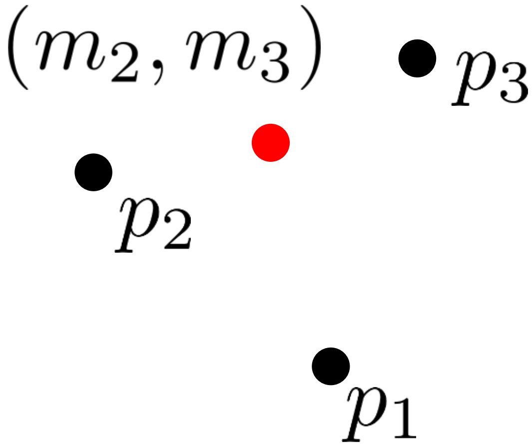
Barycentric Coordinates



Properties:

- $0 \leq m_i \leq 1$
- $0 \leq m_2 + m_3 \leq 1$

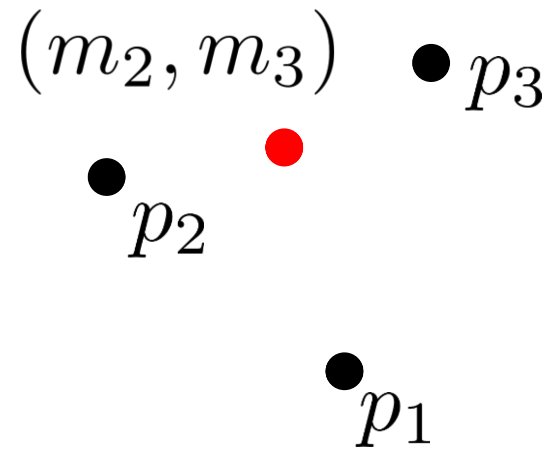
Barycentric Coordinates



Properties:

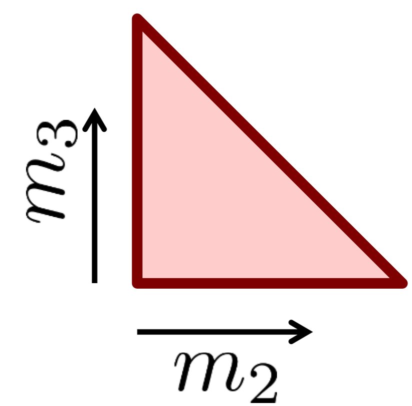
- $0 \leq m_i \leq 1$
- $0 \leq m_2 + m_3 \leq 1$
- corners are (0,0), (1,0), (0,1)
- unique for any inside point

Barycentric Coordinates

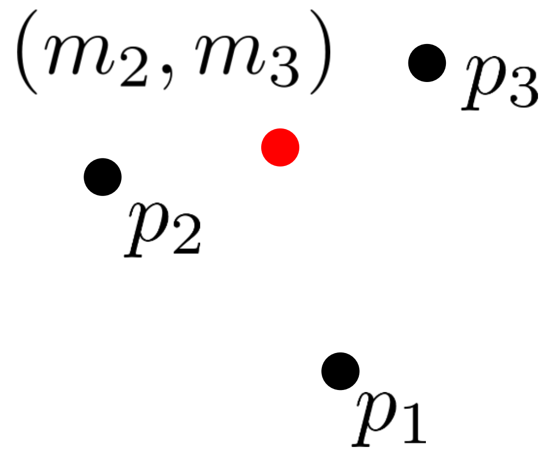


Properties:

- $0 \leq m_i \leq 1$
- $0 \leq m_2 + m_3 \leq 1$
- corners are (0,0), (1,0), (0,1)
- unique for any inside point

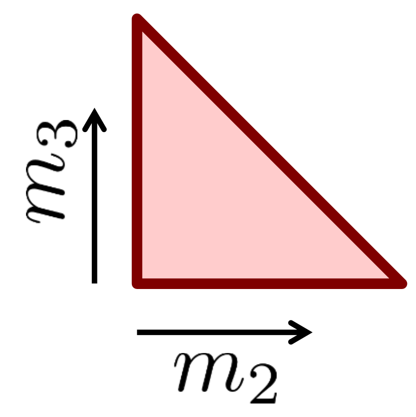


Barycentric Coordinates



Properties:

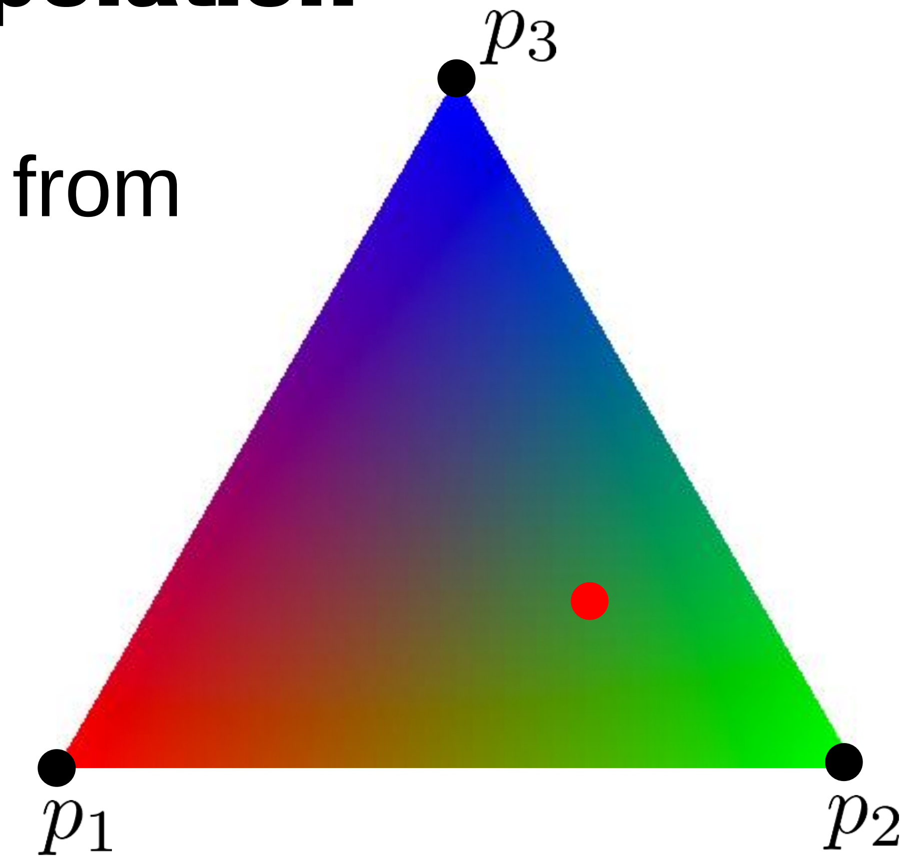
- $0 \leq m_i \leq 1$
- $0 \leq m_2 + m_3 \leq 1$
- corners are (0,0), (1,0), (0,1)
- unique for any inside point



Why do we care?

Barycentric Interpolation

Extends any function from corners to triangle

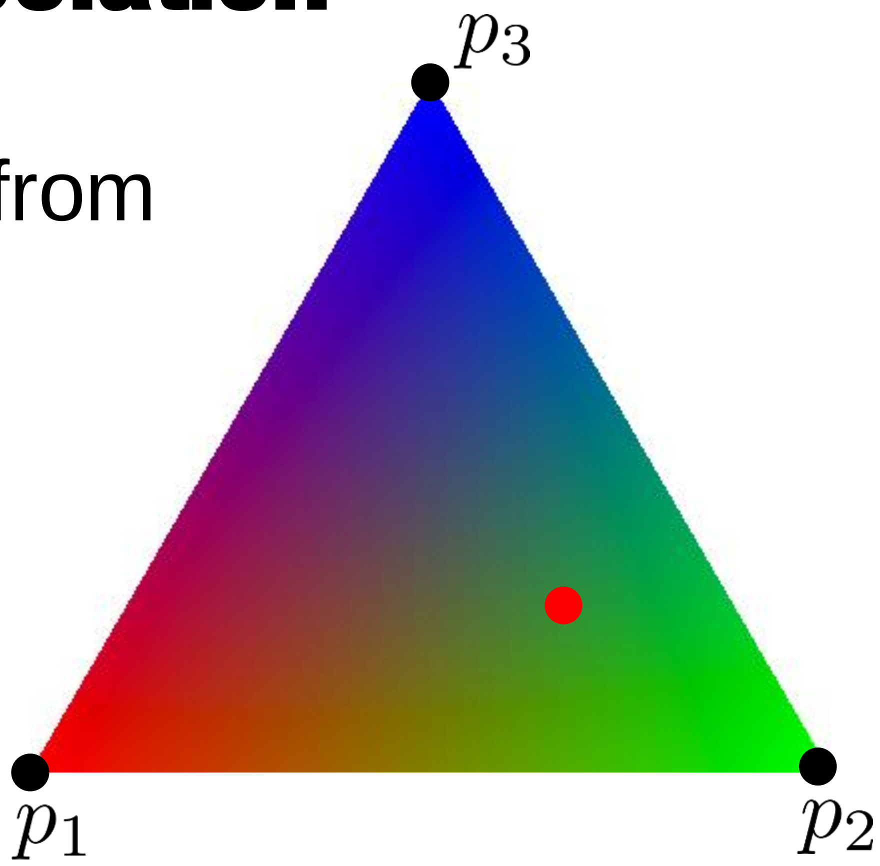


$$f_c = (1 - m_2 - m_3)f_1 + m_2 f_2 + m_3 f_3$$

Barycentric Interpolation

Extends any function from corners to triangle

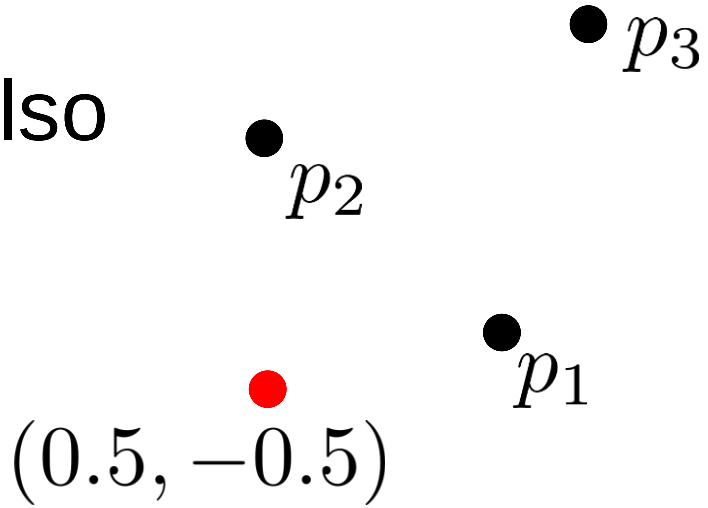
- colors
- normals
- whatever



$$f_c = (1 - m_2 - m_3) f_1 + m_2 f_2 + m_3 f_3$$

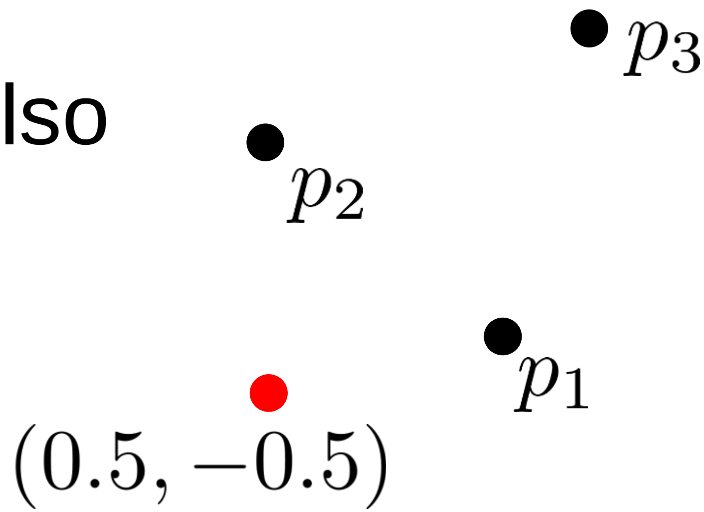
Negative Barycentric Coordinates

Points outside triangle also have coords



Negative Barycentric Coordinates

Points outside triangle also have coords

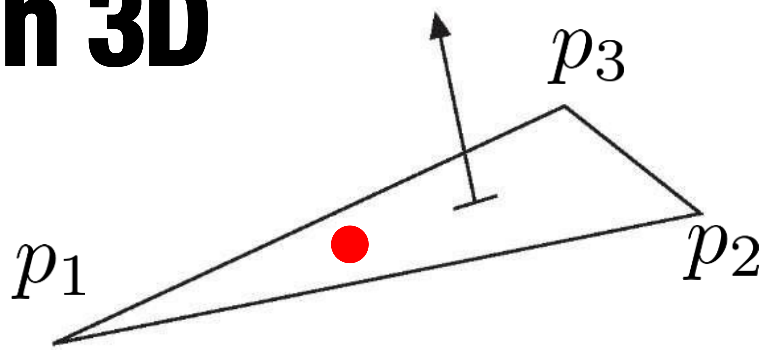


Alternate inside-triangle check:

- compute barycentric coords
- check they're valid

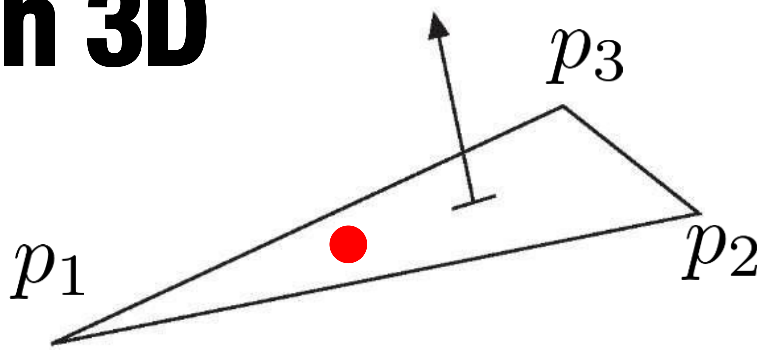
Barycentric Coords in 3D

Given c **in plane of tri:**
find coords with



$$c - p_1 = m_2(p_2 - p_1) + m_3(p_3 - p_1)$$

Barycentric Coords in 3D

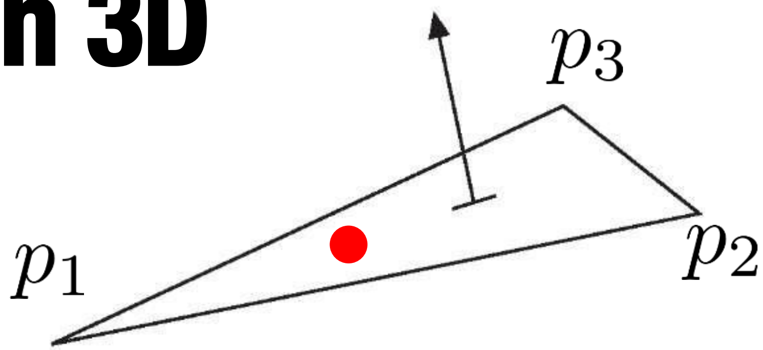


Given c **in plane of tri:**
find coords with

$$c - p_1 = m_2(p_2 - p_1) + m_3(p_3 - p_1)$$

Problem: too many equations!!

Barycentric Coords in 3D



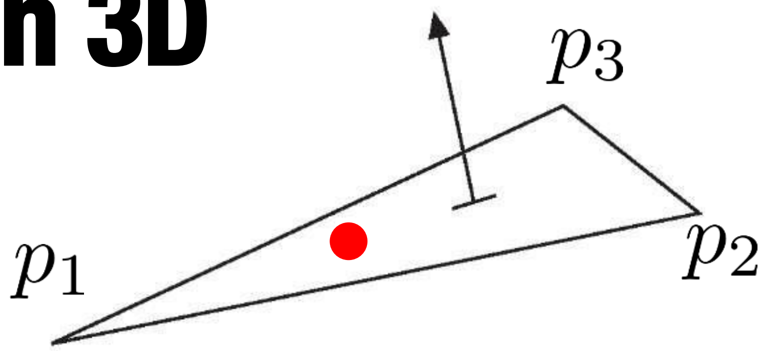
Given c **in plane of tri:**
find coords with

$$c - p_1 = m_2(p_2 - p_1) + m_3(p_3 - p_1)$$

Problem: too many equations!!

Can we eliminate one of the variables?

Barycentric Coords in 3D



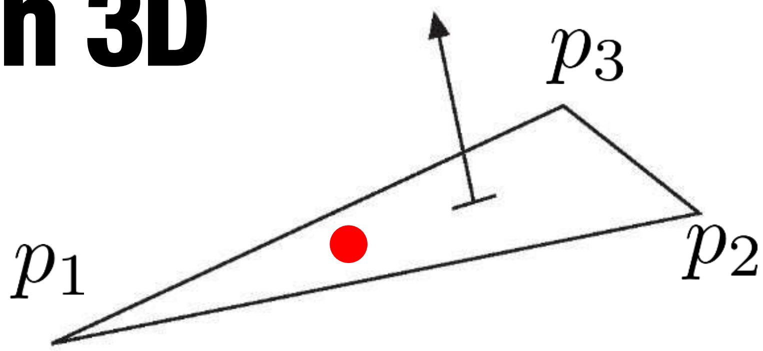
Given c **in plane of tri:**
find coords with

$$c - p_1 = m_2(p_2 - p_1) + m_3(p_3 - p_1)$$

$$(p_3 - p_1) \times (c - p_1) = m_2(p_3 - p_1) \times (p_2 - p_1)$$

Both sides vectors in normal **direction**

Barycentric Coords in 3D



Given c **in plane of tri:** find coords with

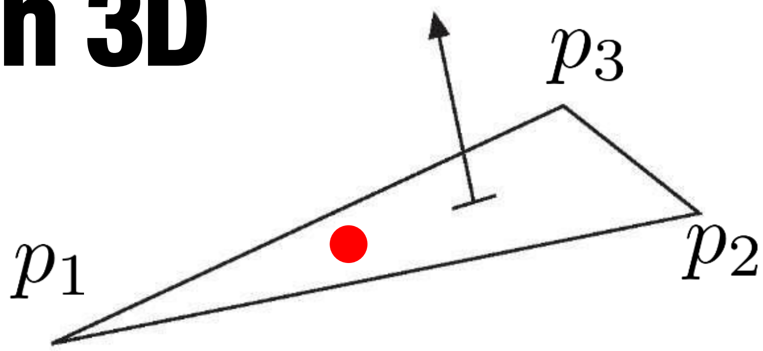
$$c - p_1 = m_2(p_2 - p_1) + m_3(p_3 - p_1)$$

$$(p_3 - p_1) \times (c - p_1) = m_2(p_3 - p_1) \times (p_2 - p_1)$$

Both sides vectors in normal **direction**

$$[(p_3 - p_1) \times (c - p_1)] \cdot \hat{n} = m_2 \left[(p_3 - p_1) \times (p_2 - p_1)\right] \cdot \hat{n}$$

Barycentric Coords in 3D



Given c **in plane of tri:**
find coords with

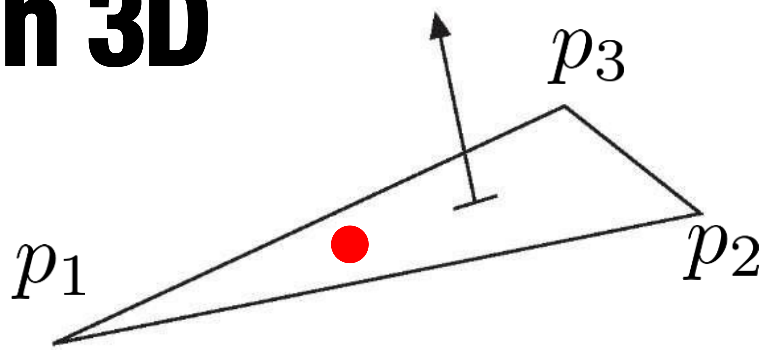
$$c - p_1 = m_2(p_2 - p_1) + m_3(p_3 - p_1)$$

$$(p_3 - p_1) \times (c - p_1) = m_2(p_3 - p_1) \times (p_2 - p_1)$$

Both sides vectors in normal **direction**

$$[(p_3 - p_1) \times (c - p_1)] \cdot \hat{n} = m_2 \left[(p_3 - p_1) \times (p_2 - p_1)\right] \cdot \hat{n}$$
$$[(p_2 - p_1) \times (c - p_1)] \cdot \hat{n} = m_3 \left[(p_2 - p_1) \times (p_3 - p_1)\right] \cdot \hat{n}$$

Barycentric Coords in 3D

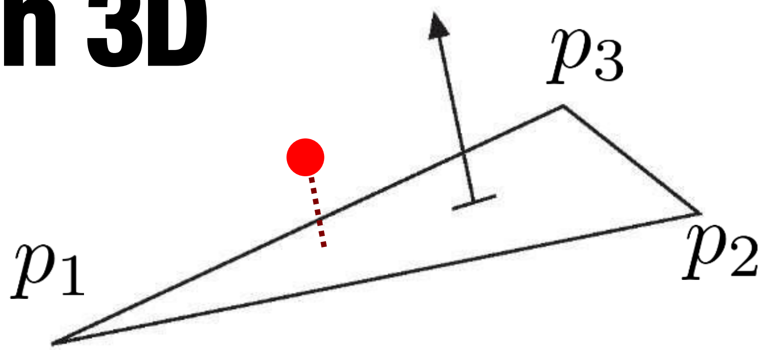


Given c **in plane of tri:**
find coords with

$$m_2 = \frac{[(p_3-p_1)\times(c-p_1)]\cdot\hat{n}}{[(p_3-p_1)\times(p_2-p_1)]\cdot\hat{n}}$$

$$m_3 = \frac{[(p_2-p_1)\times(c-p_1)]\cdot\hat{n}}{[(p_2-p_1)\times(p_3-p_1)]\cdot\hat{n}}$$

Barycentric Coords in 3D



Given c **in plane of tri:**
find coords with

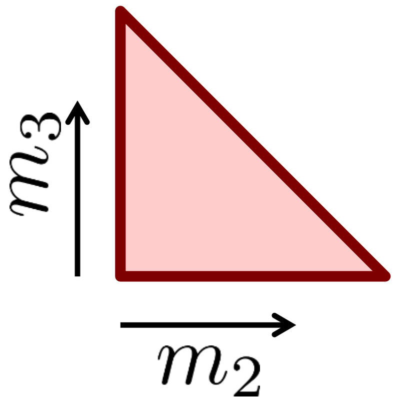
$$m_2 = \frac{[(p_3 - p_1) \times (c - p_1)] \cdot \hat{n}}{[(p_3 - p_1) \times (p_2 - p_1)] \cdot \hat{n}}$$

$$m_3 = \frac{[(p_2 - p_1) \times (c - p_1)] \cdot \hat{n}}{[(p_2 - p_1) \times (p_3 - p_1)] \cdot \hat{n}}$$

What if c is **not** in the plane of triangle?

Ray Tracing Triangles

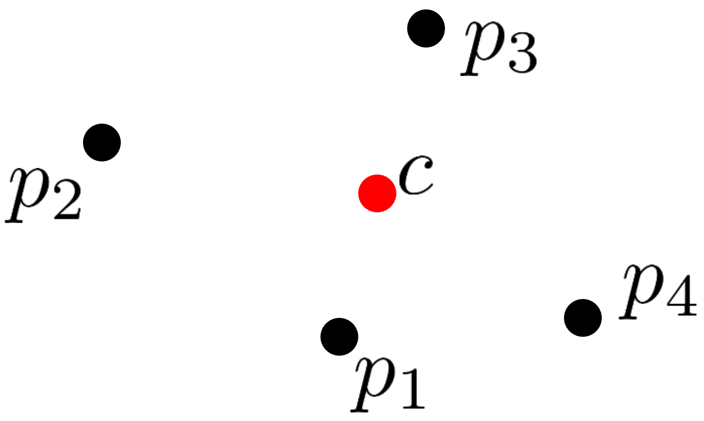
1. Find point where ray hits triangle plane
2. Calculate barycentric coordinates
3. Check coords valid



4. Linearly interpolate normals etc.
5. Shade pixel

Beyond Triangles

Much carries over…

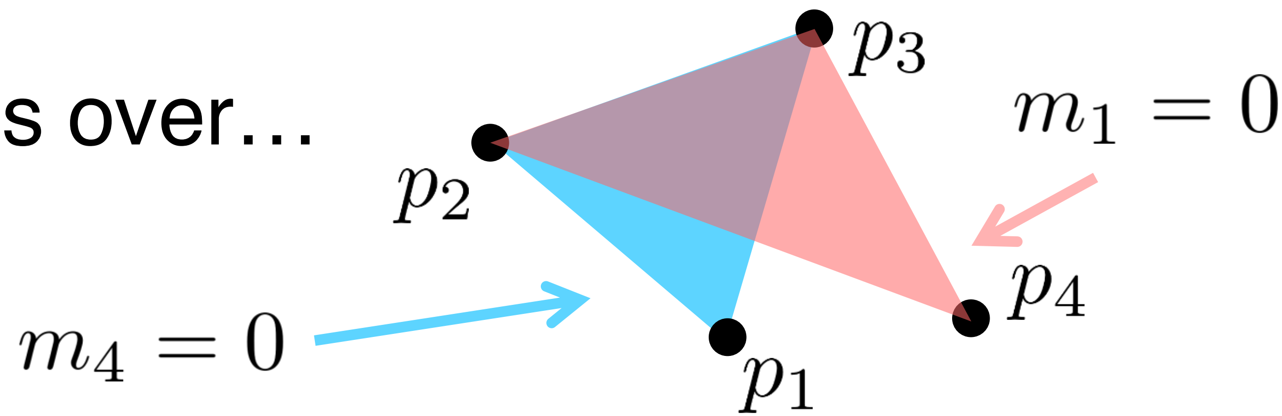


$$c = (1 - m_2 - m_3 - m_4)p_1 + m_2 p_2 + m_3 p_3 + m_4 p_4$$

Are coords still unique?

Beyond Triangles

Much carries over…

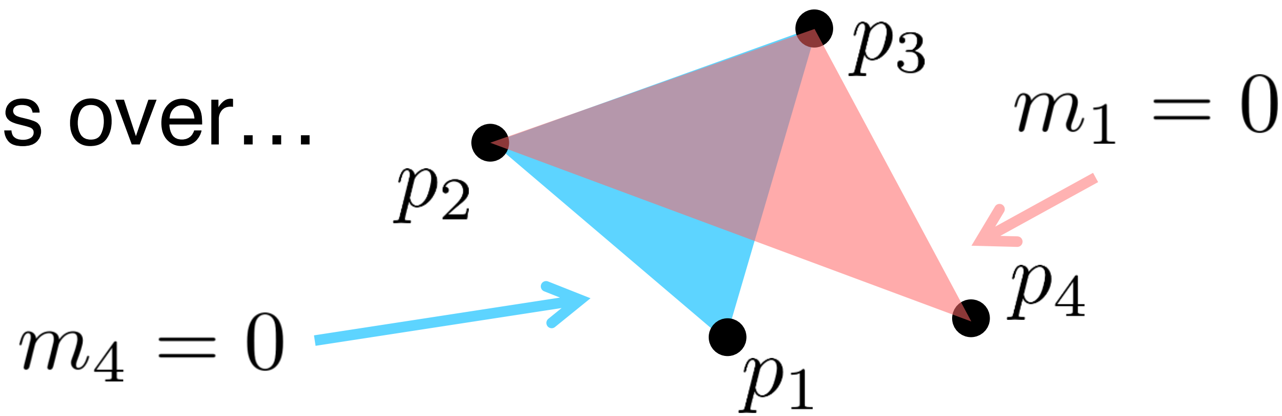


$$c = (1 - m_2 - m_3 - m_4)p_1 + m_2 p_2 + m_3 p_3 + m_4 p_4$$

Are coords still unique? No!

Beyond Triangles

Much carries over…



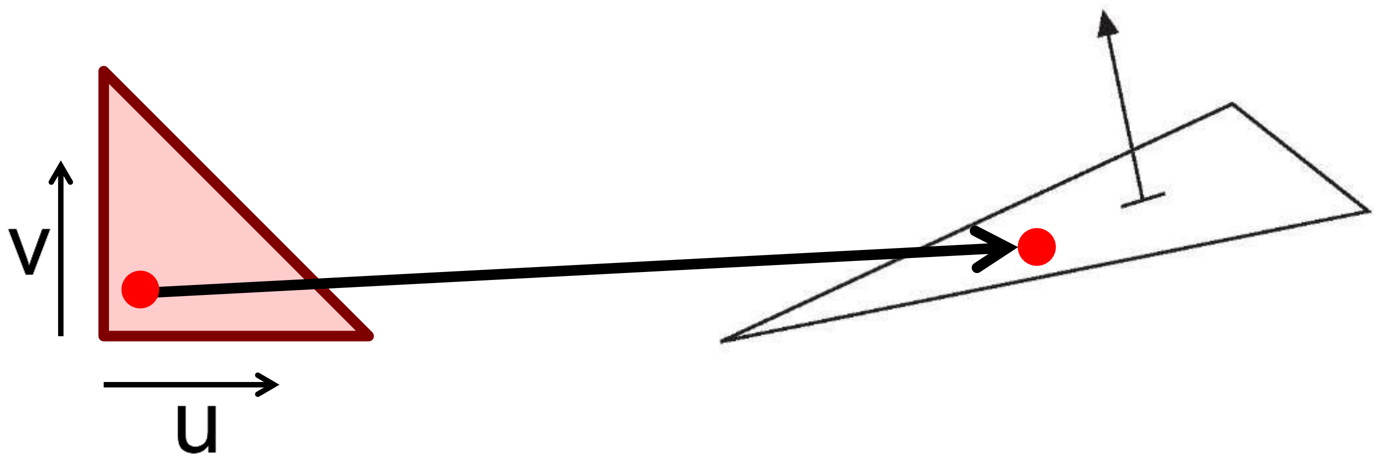
$$c = (1 - m_2 - m_3 - m_4)p_1 + m_2 p_2 + m_3 p_3 + m_4 p_4$$

Are coords still unique? No!

Many **generalized barycentric coords** schemes exist

Barycentric Coords as a Map

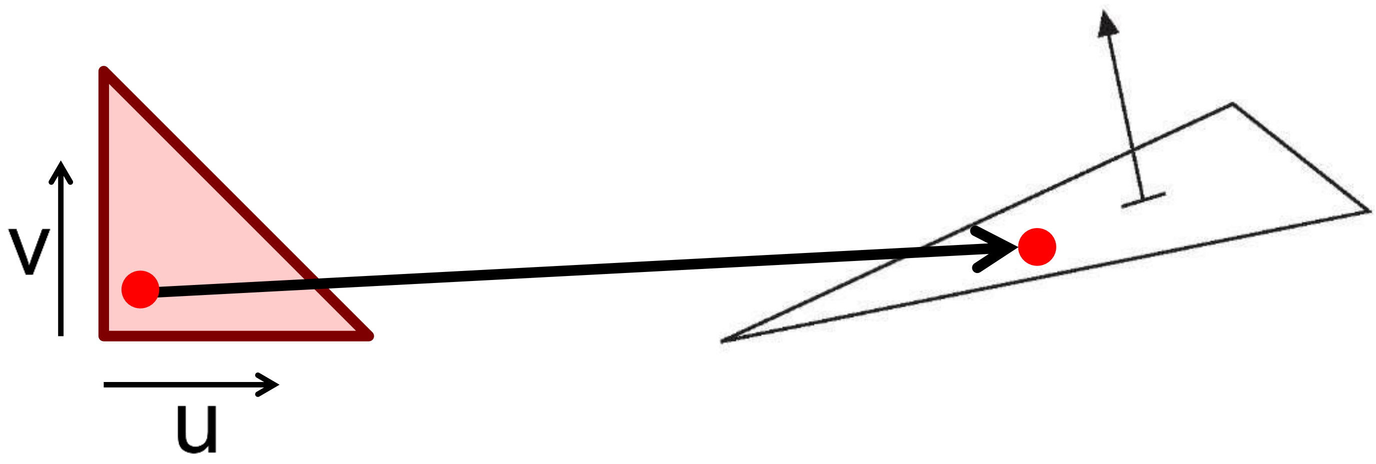
Maps from a triangle **in 2D** to 3D triangle



Called **parameterization** of triangle

Barycentric Coords as a Map

Maps from a triangle **in 2D** to 3D triangle

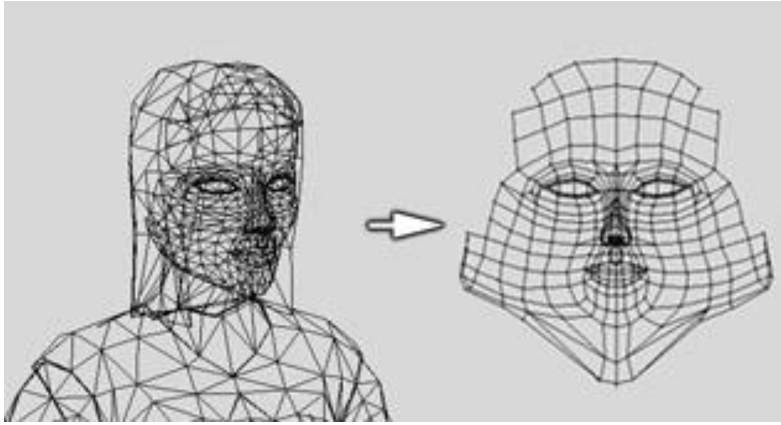


Called **parameterization** of triangle

- from now on, 2D coords are **u** and **v**

Parameterization

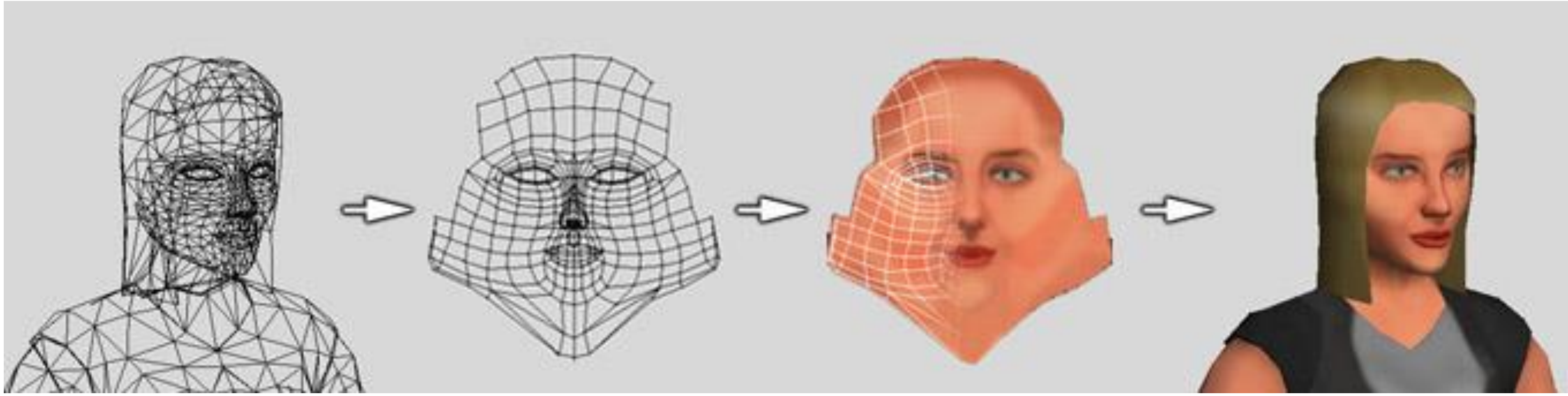
Map between **region of plane** and **arbitrary surface**



why do we want to do this?

Parameterization

Map between **region of plane** and **arbitrary surface**



Can then use parameterization to paint image on 3D surface: **texture map**

Texture Map

Parameterization == texture map
== UV coordinates
== UV unwrapping

Texture Map

Parameterization == texture map
== UV coordinates
== UV unwrapping

Usually means assigning U and V coordinates to every pixel

Texture Map

Parameterization == texture map
== UV coordinates
== UV unwrapping

Usually means assigning U and V coordinates to every pixel

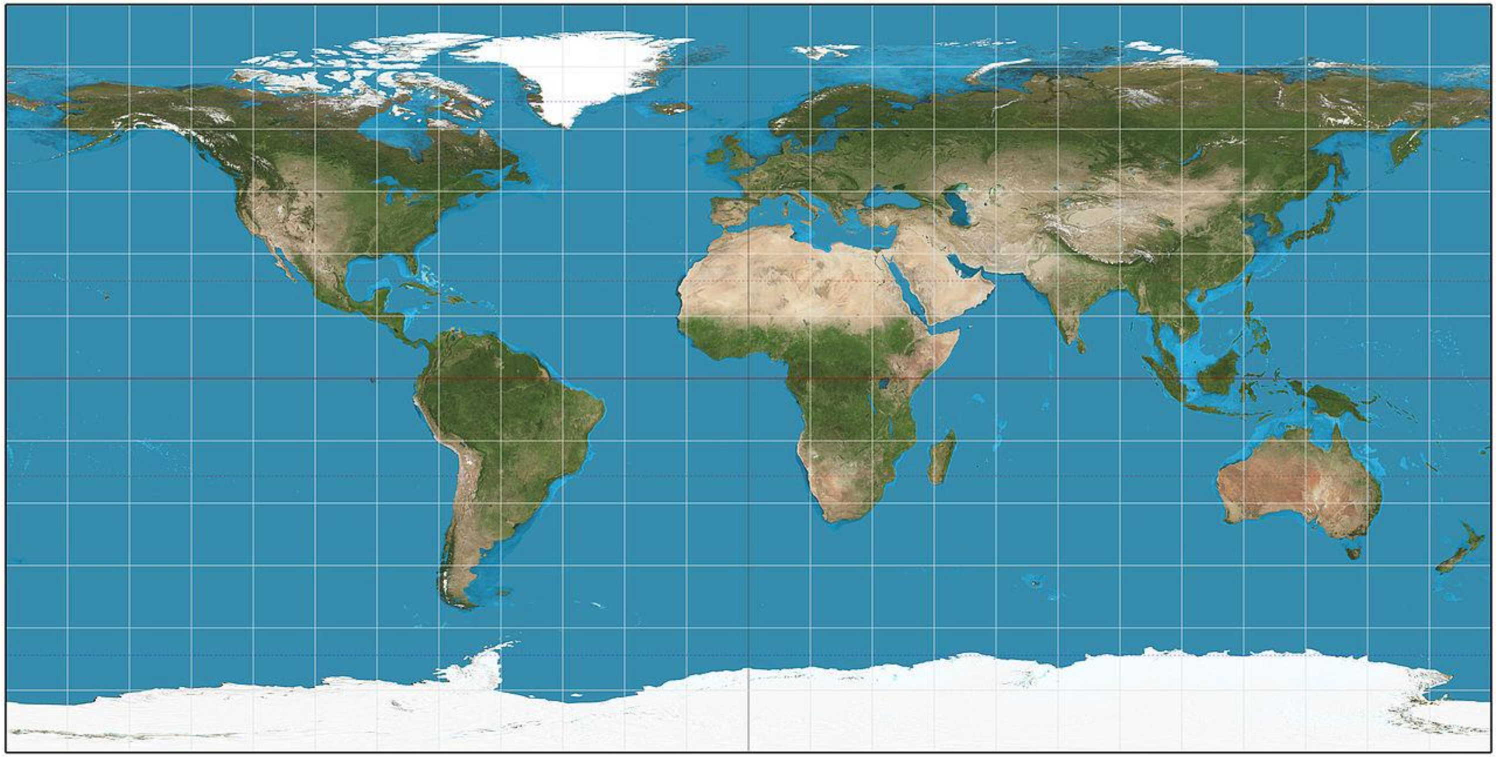
Or U and V for every vertex, then interpolate

Parameterization History

How to parameterize the earth (sphere)?

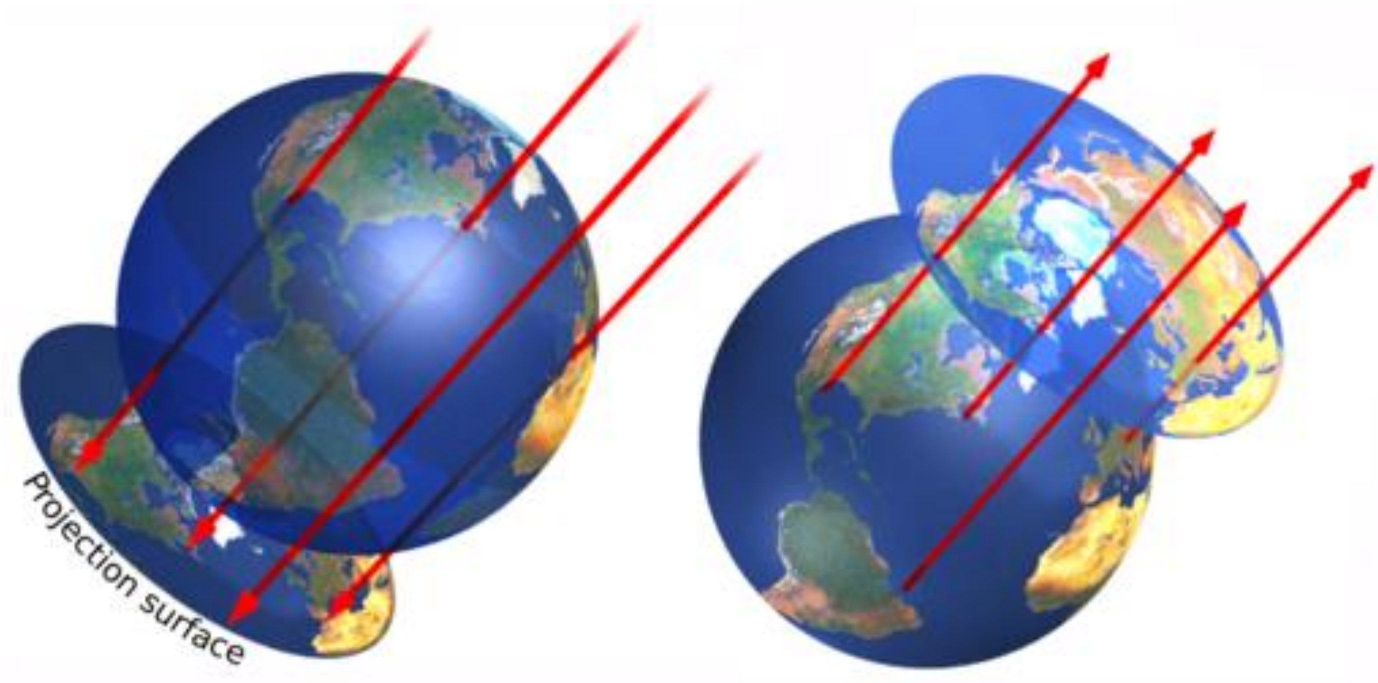
Very practical, important problem in Middle Ages…

Latitude & Longitude



Distorts areas and angles

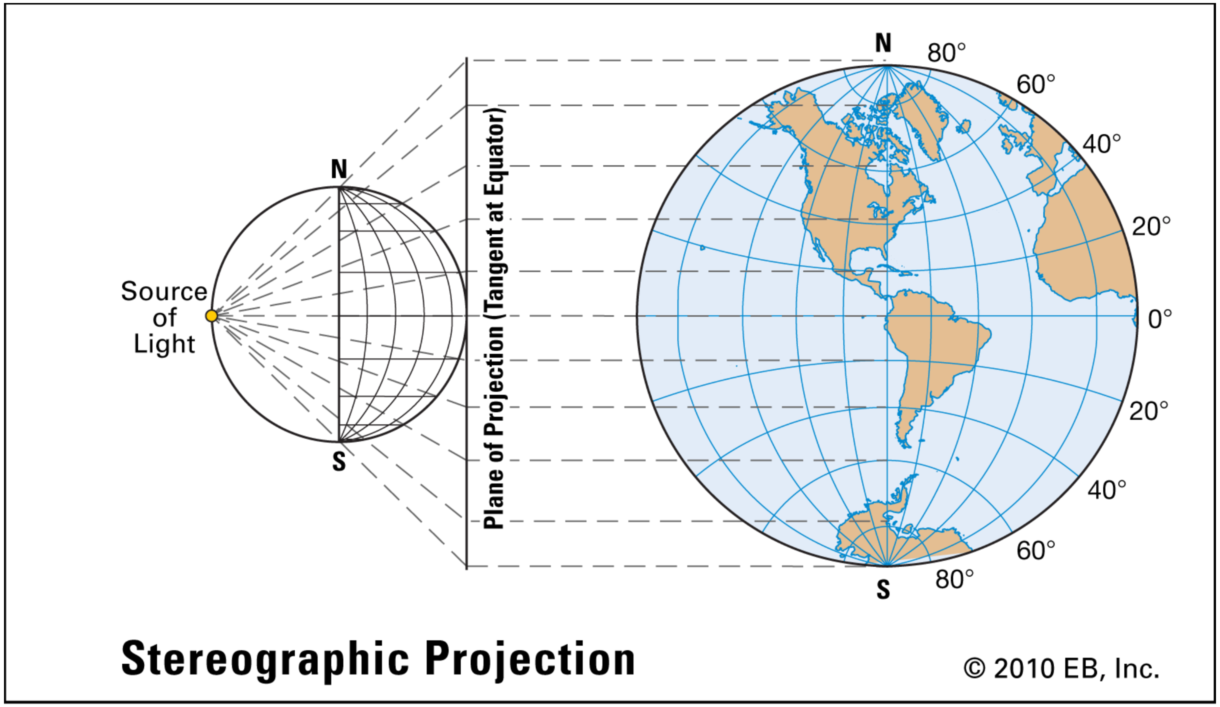
Planar Projection



Covers only half of the earth

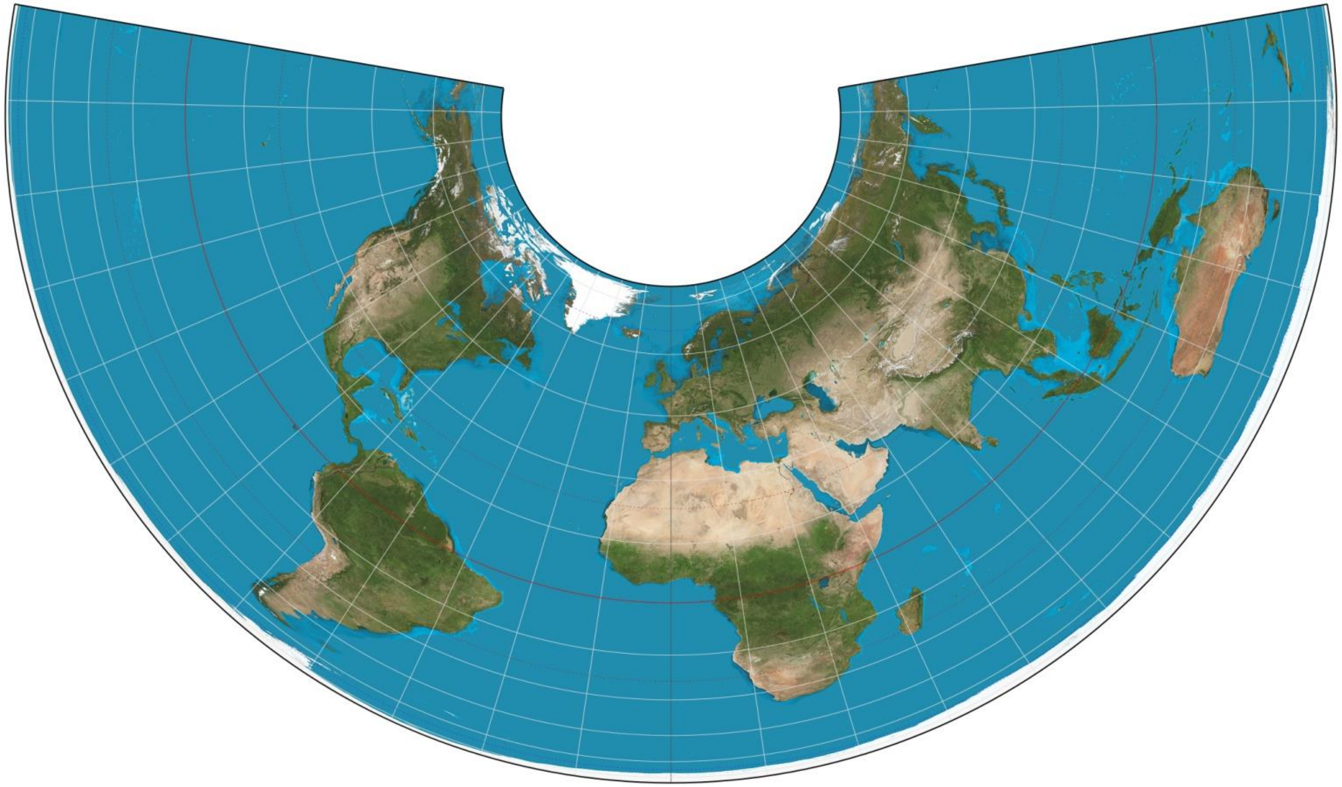
Distorts areas and angles

Stereographic Projection



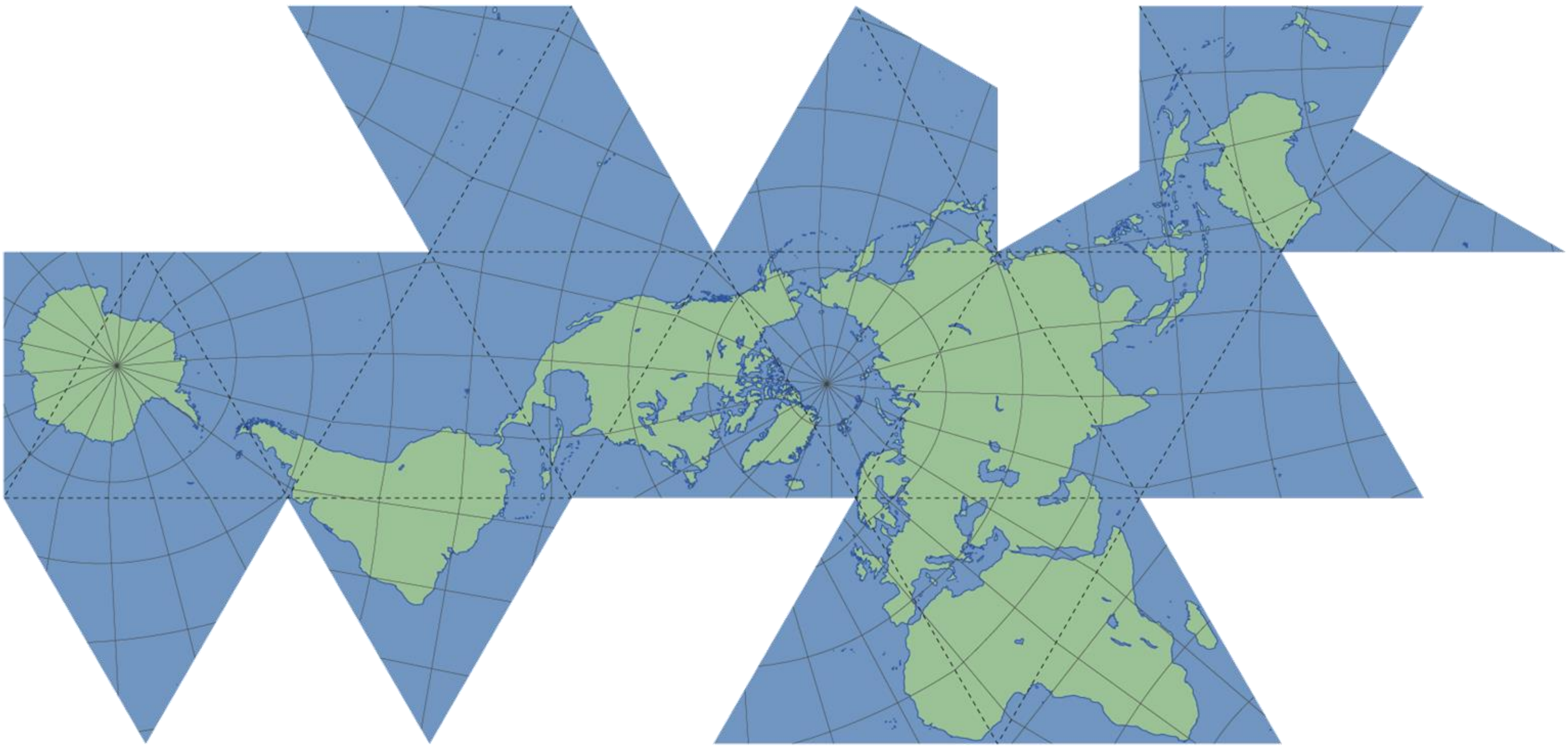
Distorts areas

Albers Projection



Preserves areas, distorts aspect ratio

Fuller Parameterization



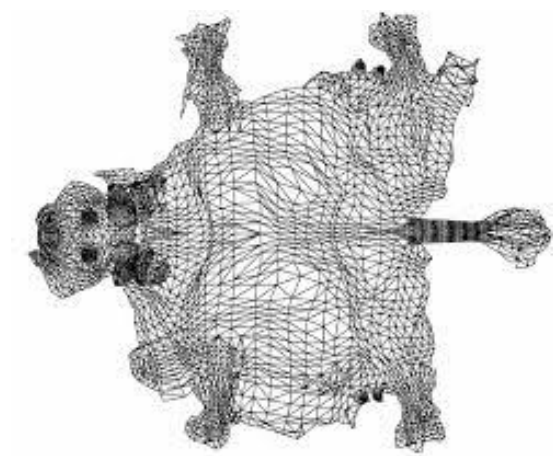
No Free Lunch

Every parameterization of the earth either:

- distorts areas
- distorts distances
- distorts angles

Good Parameterizations

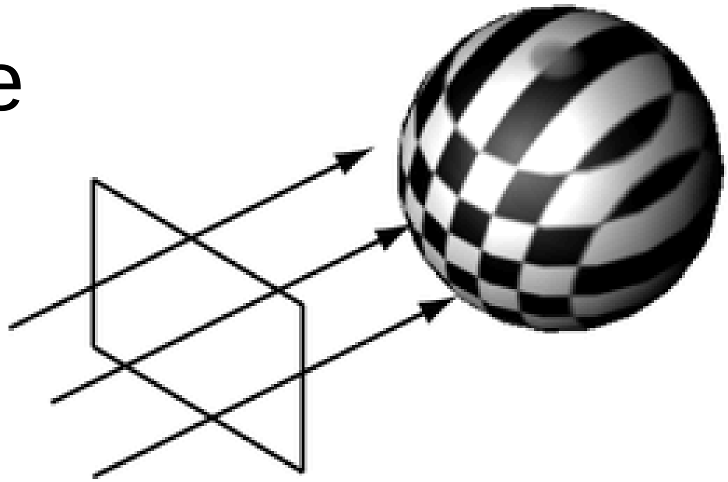
- low area distortion
- low angle distortion
- no obvious seams
- one piece



Soup Parameterization

Planar Parameterization

Project surface onto plane



Planar Parameterization

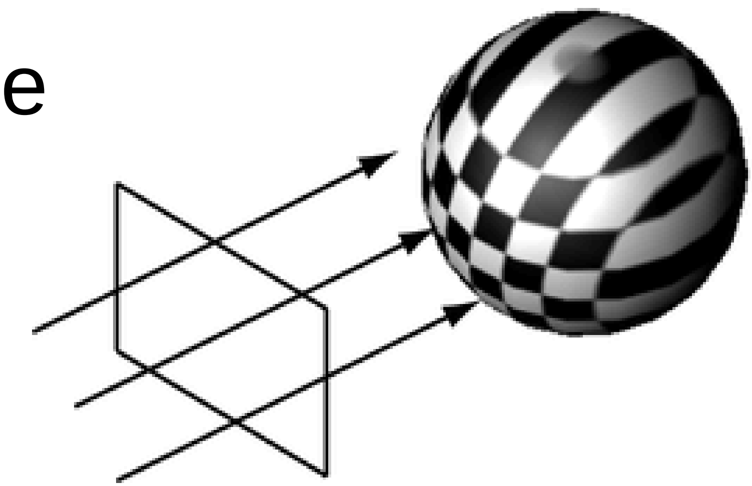
Project surface onto plane

- quite useful in practice

Planar Parameterization

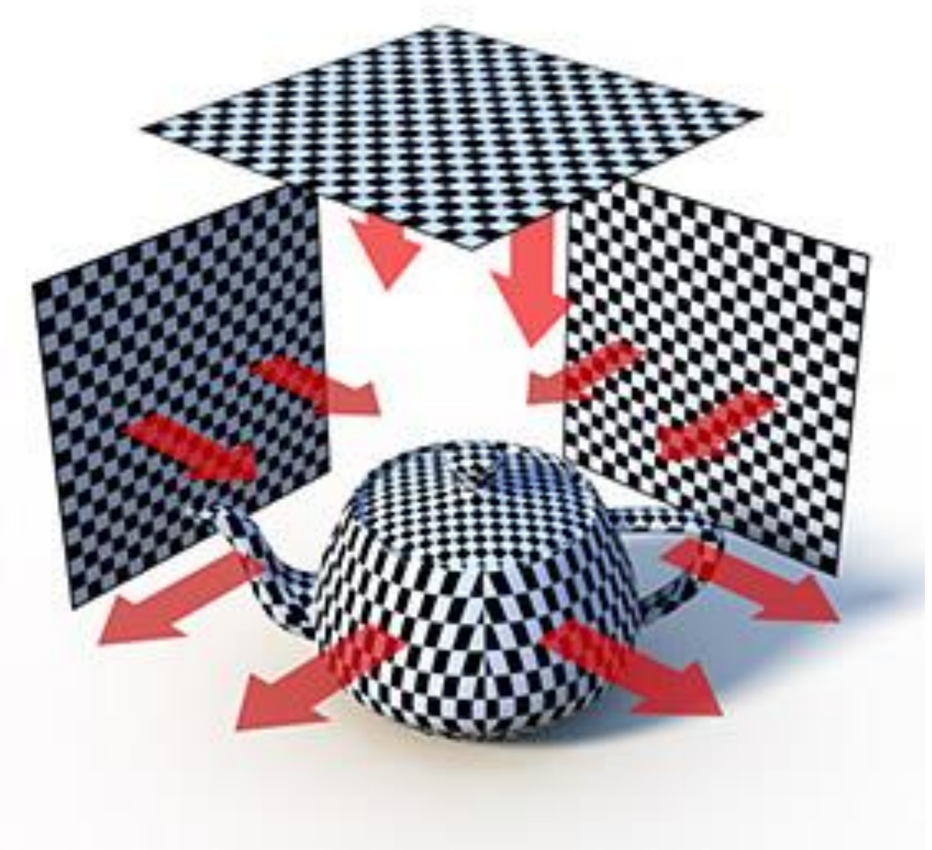
Project surface onto plane

- quite useful in practice
- only partial coverage
- bad distortion when normals perpendicular

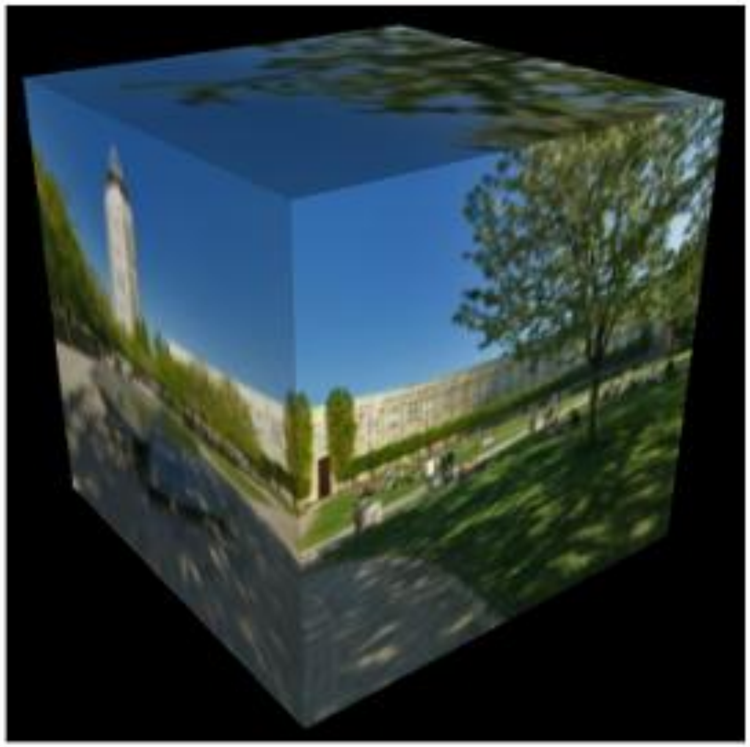


Planar Parameterization

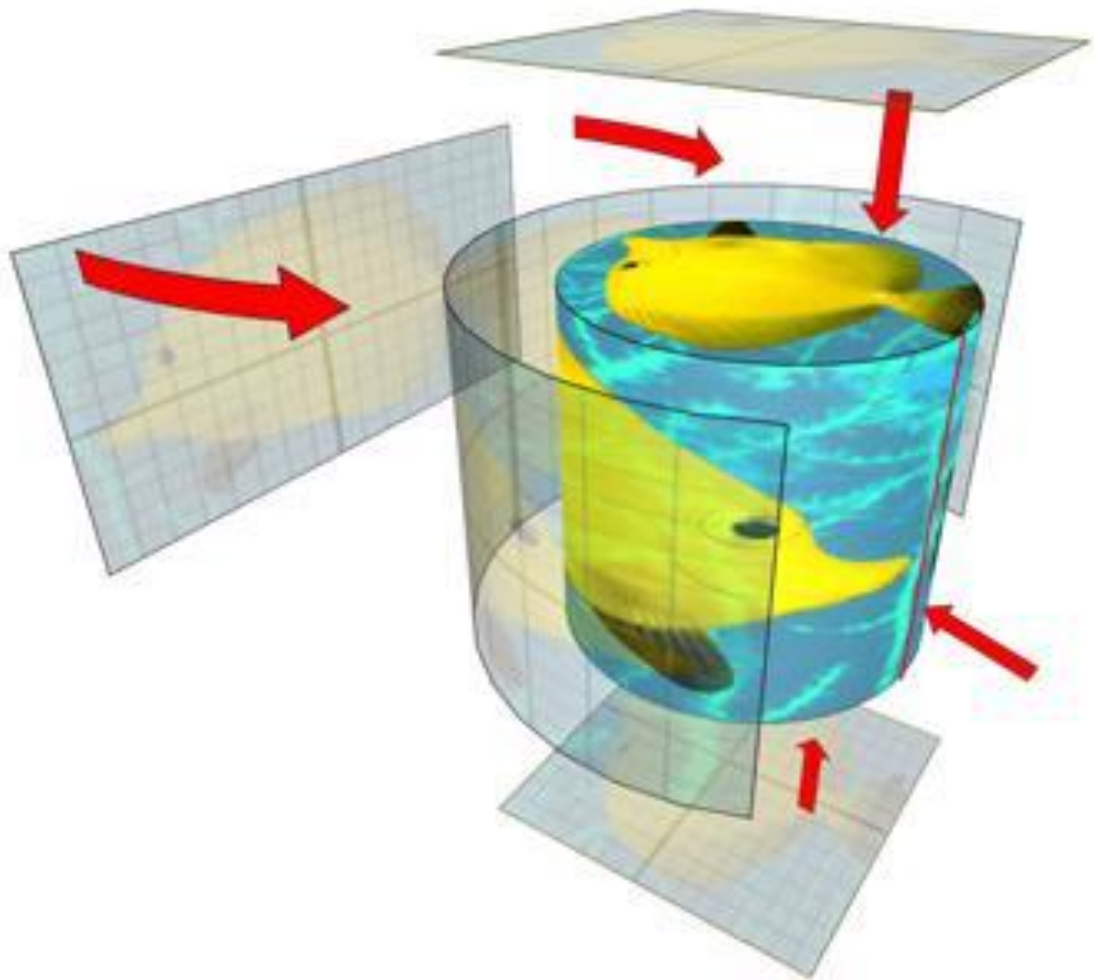
In practice: combine multiple views



Cube Map



Cylindrical Parameterization



Conformal Parameterization

Conformal = angle-preserving

Conformal Parameterization

Conformal = angle-preserving

Riemann mapping theorem

- can map any surface conformally

Conformal Parameterization

Conformal = angle-preserving

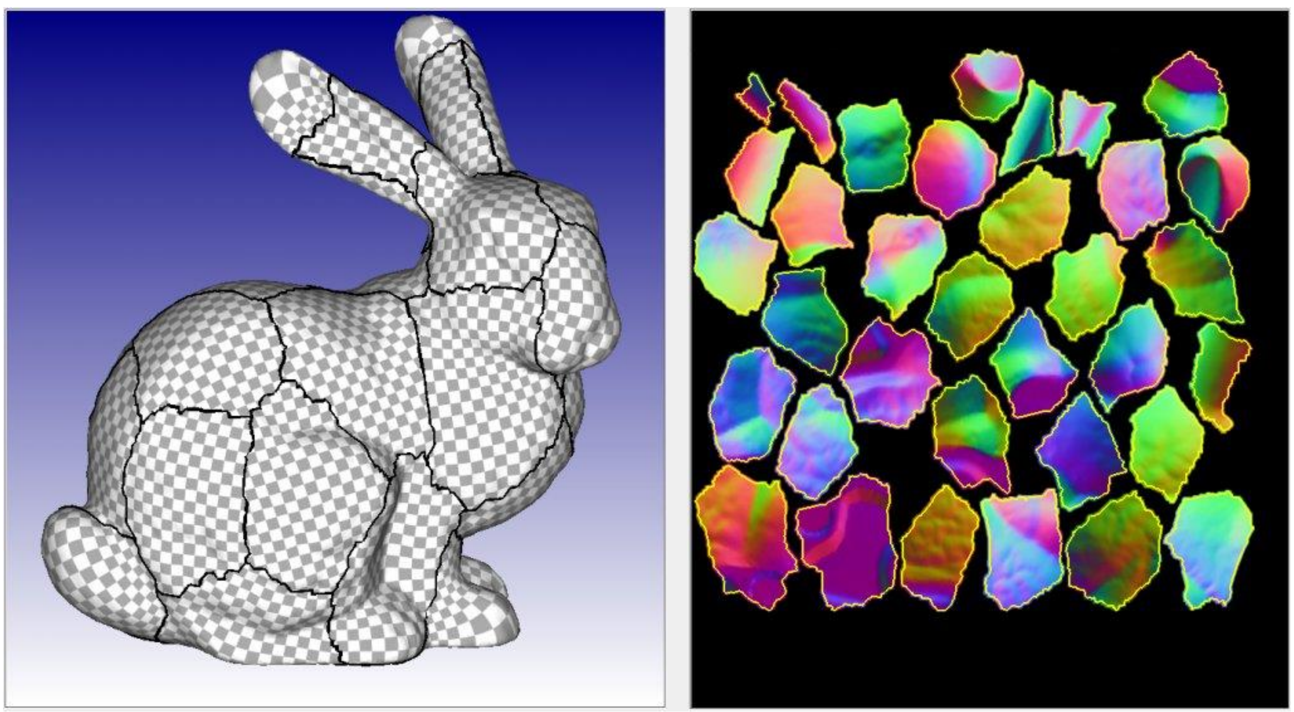
Riemann mapping theorem

- can map any surface conformally

Area distortion can be bad

Texture Atlas

Break up surface into easy pieces, parameterize separately



Texture Atlas

Some automatic methods exist…

but often artists hand-paint UV coords

Projection Mapping

Projection Mapping

Scan 3D geometry, compute texture map

Then, project anything you want on object