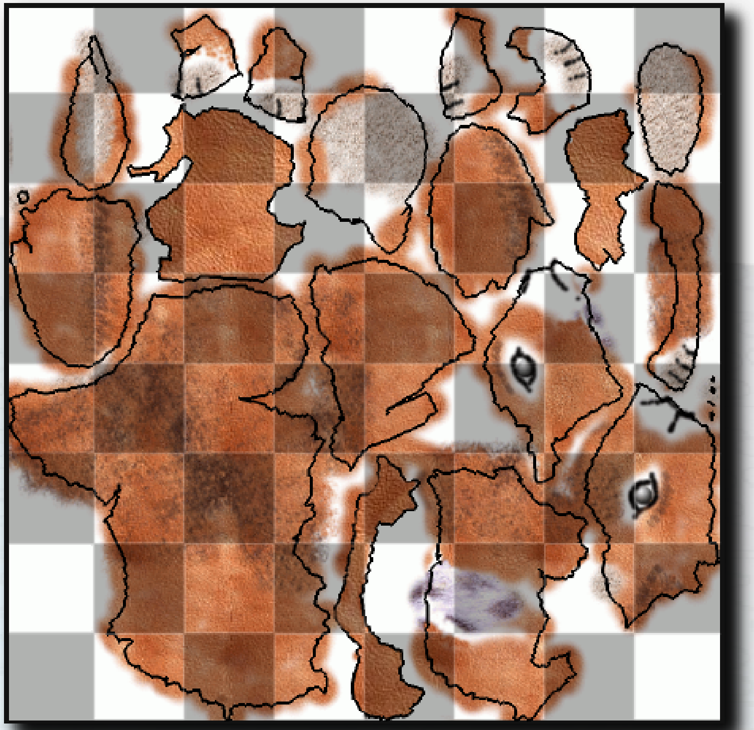
Surface Parameterization



Julien Tierny

tierny@telecom-paristech.fr

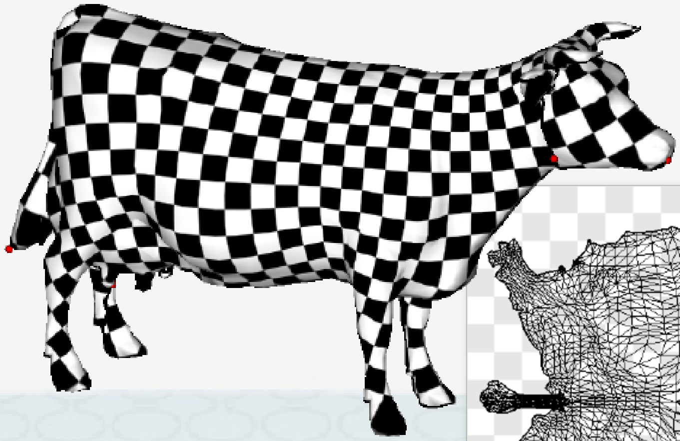
Office C04

March 14, 2012

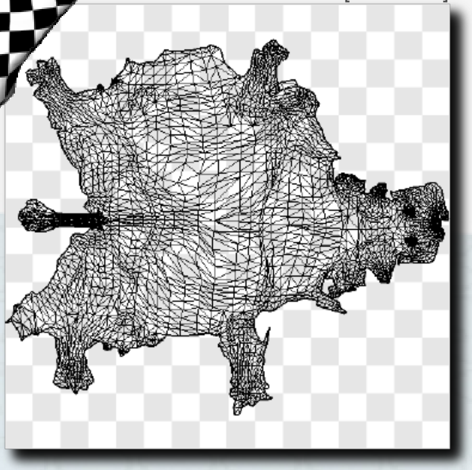
Summary

- What? What for?
- Preliminary background
- A simple algorithm
- Local parameterization
  - Least Squares Conformal Maps [Levy et al. 2002]
- Global parameterization
  - Curvature prescription, circle packing and metric optimization [Kharevych et al. 2006, Jin et al. 2008, Ben Chen et al. 2008]
- Perspectives
  - Quadrangulation, cross parameterization, volume parameterization

What? What for?

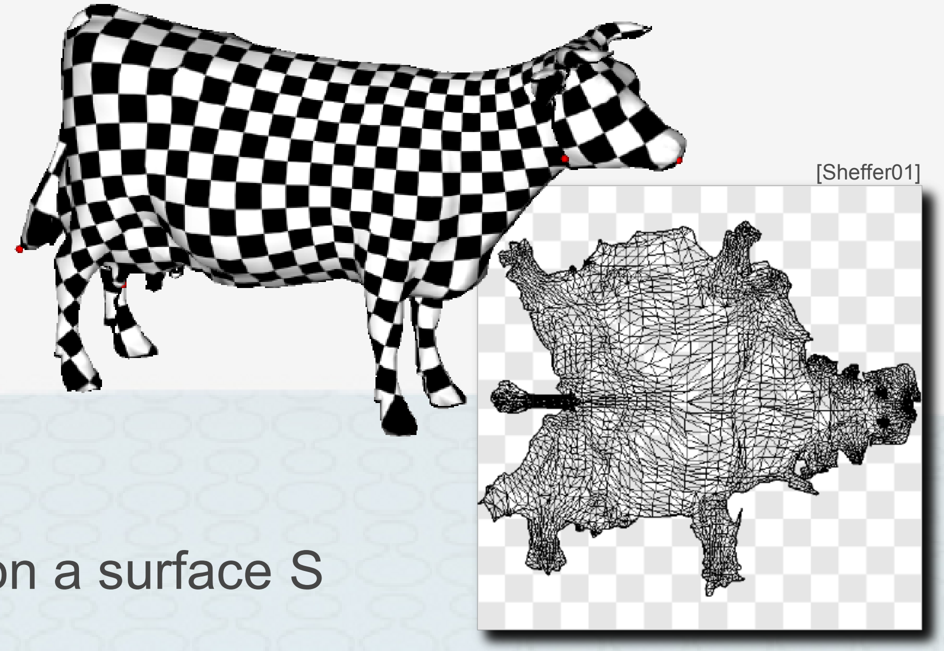


[Sheffer01]



- Construct a coordinate system on a surface S
- Find a bijective mapping to *some* reference domain D
  - $\phi : \mathcal{S} \to \mathcal{D}$
  - $\mathcal{D} \subset \mathbb{R}^2, \mathcal{D} = \mathbb{S}^2$, etc.
  - *Reverse-engineering* the manifold

What? What for?



- Construct a coordinate system on a surface S
- Find a bijective mapping to *some* reference domain D
  - $\phi : \mathcal{S} \rightarrow \mathcal{D}$
  - $\mathcal{D} \subset \mathbb{R}^2, \mathcal{D} = \mathbb{S}^2$, etc.
  - *Reverse-engineering* the manifold
- Why?
  - **Texture mapping** (*"historical"*)
  - Signal processing on surfaces (bump maps, transfer, etc.)
  - Recovers a structure on an unstructured representation
  - Surface quadrangulation (animation, simulation, etc.)

Texture mapping in a nutshell

Texture mapping in a nutshell

Texture mapping in a nutshell

Raw Geometry

Texture mapping in a nutshell

Texture packing

Raw Geometry

Texture mapping in a nutshell

Texture packing

Raw Geometry

Texture mapping

Texture mapping motivations

- Mimic fine geometric details
  - Facilitate geometric modeling
  - Maintain a low memory footprint for the raw geometry
  - Re-usability of the textures

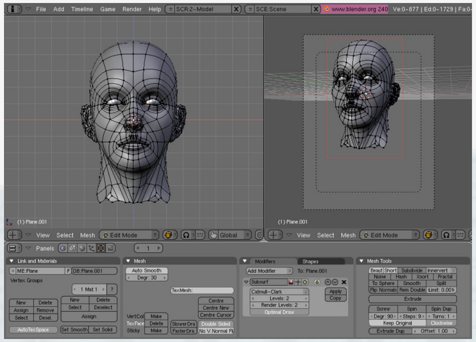
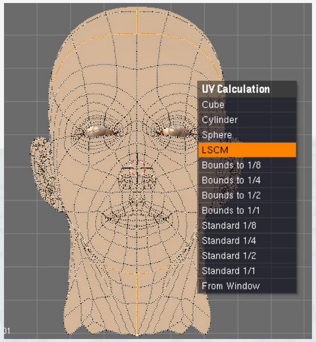
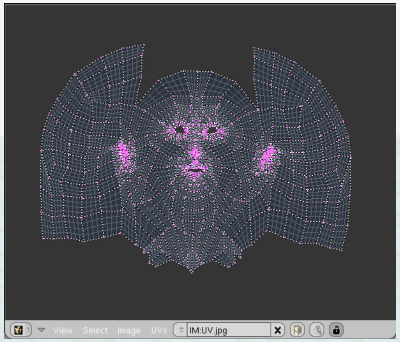
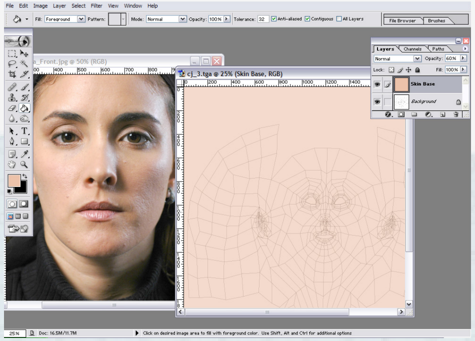
[Nintendo]

Texture mapping motivations

- Mimic fine geometric details
  - Facilitate geometric modeling
  - Maintain a low memory footprint for the raw geometry
  - Re-usability of the textures
- Mostly, interactive applications
- Typical target geometries
  - Trees
  - Buildings
  - Human faces
  - Etc.

[Nintendo]

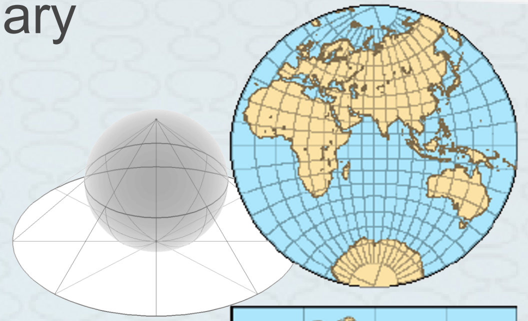
In practice

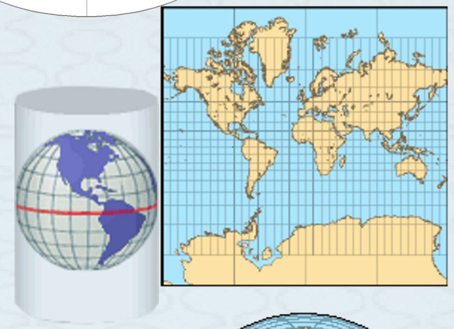





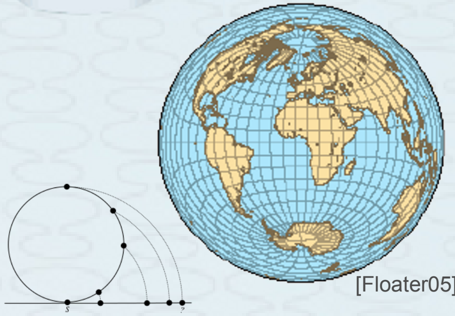
- Projections are in general not appropriate (non convex embeddings)
- Geometry unfolding
- Texture processing in the planar domain
- Challenges
  - Only developable surfaces unfold without distortion
  - Compute an unfolding map that minimizes distortion
  - What distortion are we talking about?

Preliminary background

- Unfolding stuff? We have a history!
  - Cartography: most of parameterization vocabulary
  - Orthographic projection (~2,000 BC)
  - Stereographic projection (Hipparchus 120 BC)
  - Cylindrical projection (Mercator 1594)
  - Azimuthal projection (Lambert 1777)





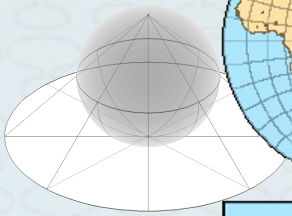


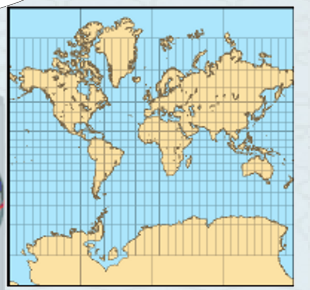
[Floater05]

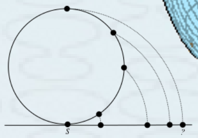
Preliminary background

- Unfolding stuff? We have a history!
  - Cartography: most of parameterization vocabulary
  - Orthographic projection (~2,000 BC)
  - Stereographic projection (Hipparchus 120 BC)
  - Cylindrical projection (Mercator 1594)
  - Azimuthal projection (Lambert 1777)
- Historical motivations
  - Facilitate navigation
  - Concerns about metric properties
    - Stereographic: preserves angles
    - Cylindrical: preserves angles + straight loxodromes
    - Azimuthal: preserves areas (national atlases)









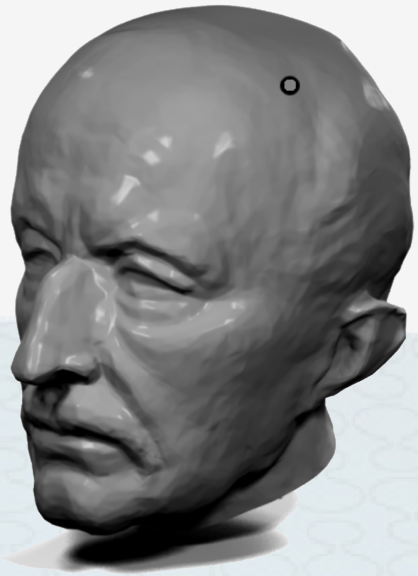
[Floater05]

Smooth setting recap

- d-manifold
  - Topological space such that every open set of it is homeomorphic to an open set of $\mathbb{R}^d$

Smooth setting recap

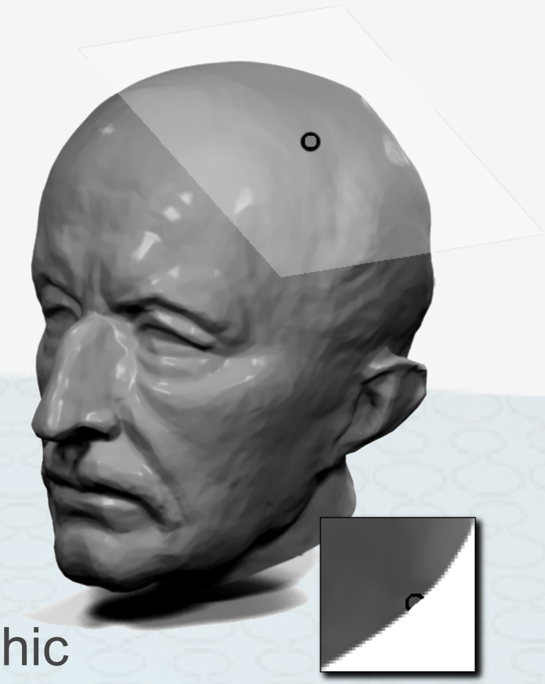
- d-manifold
  - Topological space such that every open set of it is homeomorphic to an open set of $\mathbb{R}^d$



Smooth setting recap

- d-manifold
  - Topological space such that every open set of it is homeomorphic to an open set of $\mathbb{R}^d$
- d-manifold with boundary
  - Every point has a neighborhood homeomorphic to an open of $\mathbb{R}^d$ *or* of its half space
  - Boundary: (d-1)-manifold

Smooth setting recap

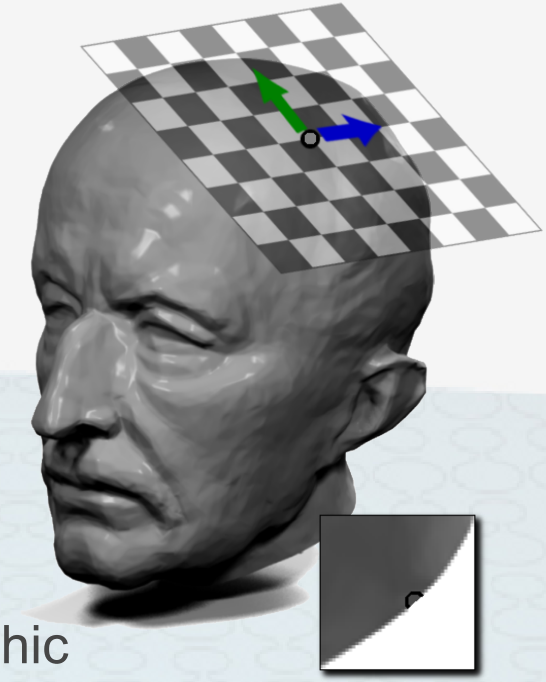


- d-manifold
  - Topological space such that every open set of it is homeomorphic to an open set of $\mathbb{R}^d$
- d-manifold with boundary
  - Every point has a neighborhood homeomorphic to an open of $\mathbb{R}^d$ *or* of its half space
  - Boundary: (d-1)-manifold
- Additional smoothness requirements (transition functions)
- Tangent plane on 2-manifolds

Smooth setting recap

- d-manifold
  - Topological space such that every open set of it is homeomorphic to an open set of $\mathbb{R}^d$
- d-manifold with boundary
  - Every point has a neighborhood homeomorphic to an open of $\mathbb{R}^d$ *or* of its half space
  - Boundary: (d-1)-manifold
- Additional smoothness requirements (transition functions)
- Tangent plane on 2-manifolds
  - Enables to pull Euclidean geometry tools for calculus on manifolds:
    - Inner product, metrics
    - Distances, angles, areas
    - Gradient, Laplace-Beltrami operator, etc.

Smooth setting recap



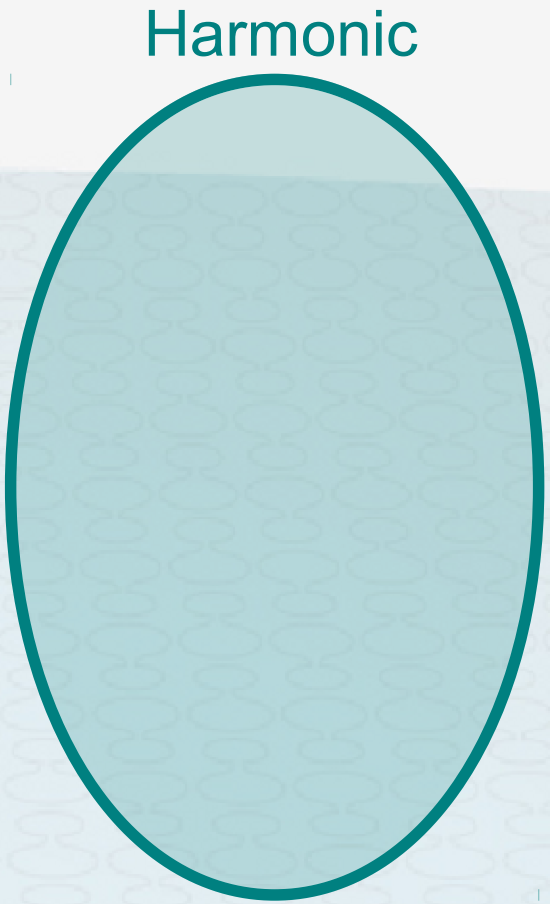
- d-manifold
  - Topological space such that every open set of it is homeomorphic to an open set of $\mathbb{R}^d$
- d-manifold with boundary
  - Every point has a neighborhood homeomorphic to an open of $\mathbb{R}^d$ *or* of its half space
  - Boundary: (d-1)-manifold
- Additional smoothness requirements (transition functions)
- Tangent plane on 2-manifolds
  - Enables to pull Euclidean geometry tools for calculus on manifolds:
    - Inner product, metrics
    - Distances, angles, areas
    - Gradient, Laplace-Beltrami operator, etc.

Map classification

- $\phi : \mathcal{S} \rightarrow \mathcal{D}$
  - $\mathcal{D}$ is parameterized with (u, v) coordinates
  - $\forall p \in \mathcal{S}, \quad \phi(p) = (u(p), v(p))$

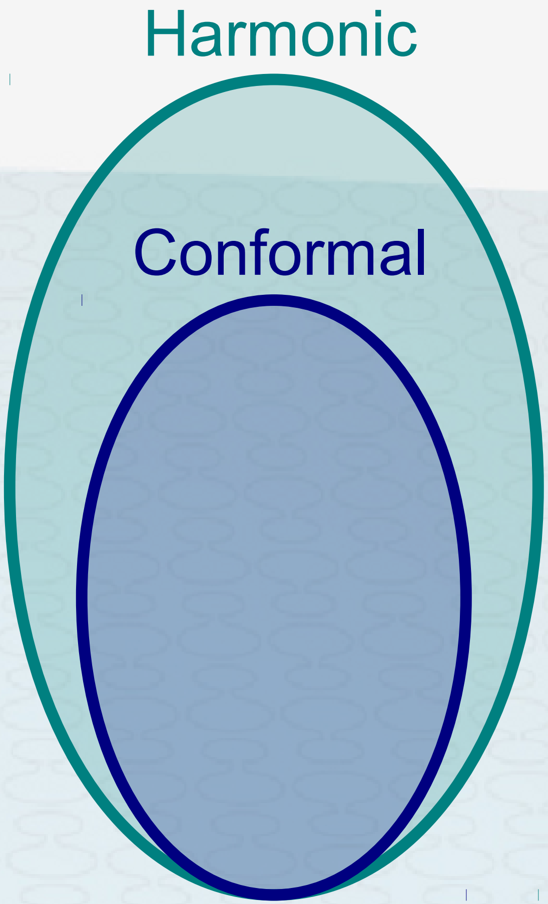
Map classification

- $\phi : \mathcal{S} \to \mathcal{D}$
  - $\mathcal{D}$ is parameterized with (u, v) coordinates
  - $\forall p \in \mathcal{S}, \quad \phi(p) = (u(p), v(p))$
- Harmonic maps
  - $\Delta u = 0, \Delta v = 0$



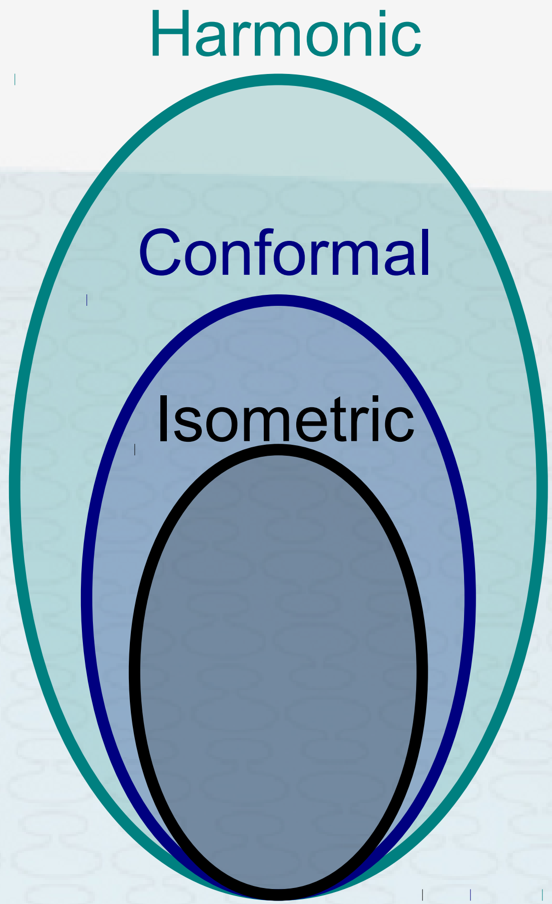
Map classification

- $\phi : \mathcal{S} \rightarrow \mathcal{D}$
  - $\mathcal{D}$ is parameterized with (u, v) coordinates
  - $\forall p \in \mathcal{S}, \quad \phi(p) = (u(p), v(p))$
- Harmonic maps
  - $\Delta u = 0, \Delta v = 0$
- Conformal maps
  - $||\nabla u|| = ||\nabla v||$
  - $\nabla u \cdot \nabla v = 0$
  - Locally isotropic: they map circles to circles
  - Preserve angles locally



Map classification

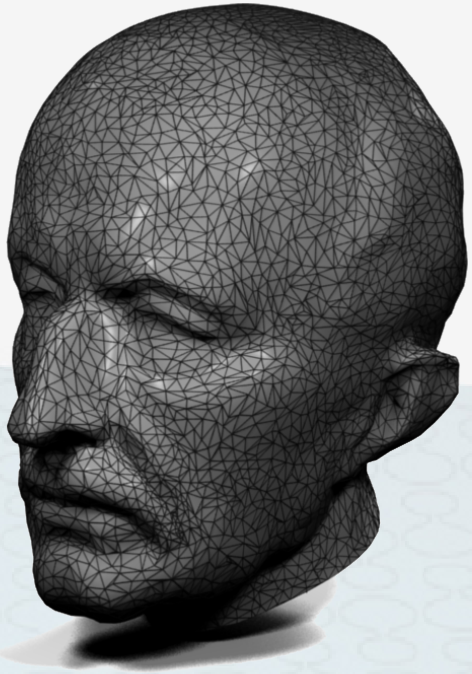
- $\phi : \mathcal{S} \to \mathcal{D}$
  - $\mathcal{D}$ is parameterized with (u, v) coordinates
  - $\forall p \in \mathcal{S}, \quad \phi(p) = (u(p), v(p))$
- Harmonic maps
  - $\Delta u = 0, \Delta v = 0$
- Conformal maps
  - $||\nabla u|| = ||\nabla v||$
  - $\nabla u \cdot \nabla v = 0$
  - Locally isotropic: they map circles to circles
  - Preserve angles locally
- Isometric maps
  - Conformal maps with zero area distortion
  - Preserve lengths



Discrete setting recap (1/2)

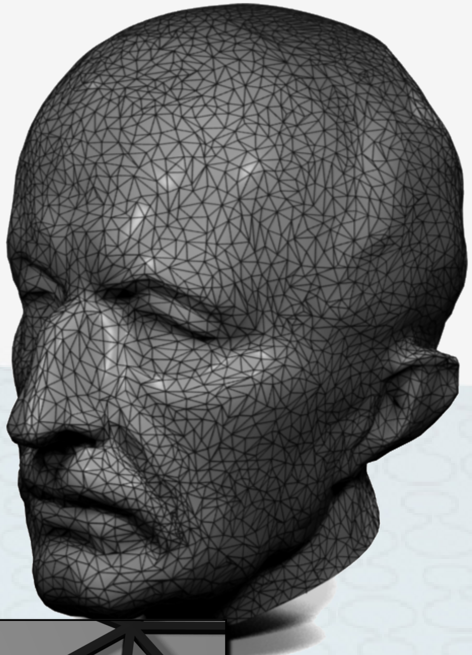
- Simplicial complex $\mathcal{K}$
  - d-simplex: convex hull of (d+1) affinely independent points in $\mathbb{R}^n$ with 0 $\leq$ d $\leq$ n
  - Vertex (0), edge (1), triangle (2)
  - Face of a d-simplex: simplex defined by a non empty subset of its d+1 points
  - $\mathcal{K}$: Collection of simplices, such that every face of a simplex is in $\mathcal{K}$ and any two simplices intersect in a common face or not at all.

Discrete setting recap (2/2)

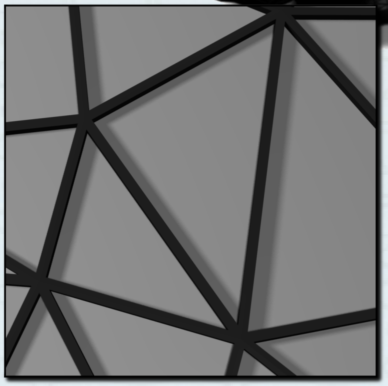


- Triangulation of a manifold $\mathcal{M}$
  - $\mathcal{K}$, such that the union of its simplices is homeomorphic to $\mathcal{M}$

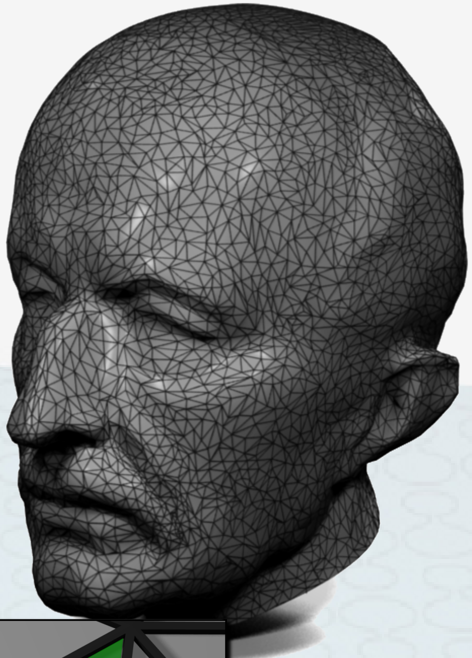
Discrete setting recap (2/2)



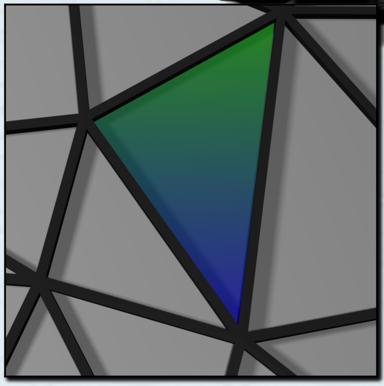
- Triangulation of a manifold $\mathcal{M}$
  - $\mathcal{K}$, such that the union of its simplices is homeomorphic to $\mathcal{M}$
  - PL 2-manifold:
    - Surface triangulation



Discrete setting recap (2/2)

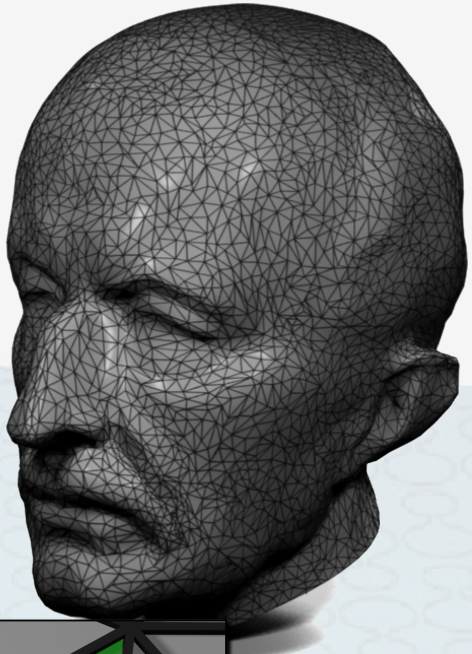


- Triangulation of a manifold $\mathcal{M}$
  - $\mathcal{K}$, such that the union of its simplices is homeomorphic to $\mathcal{M}$

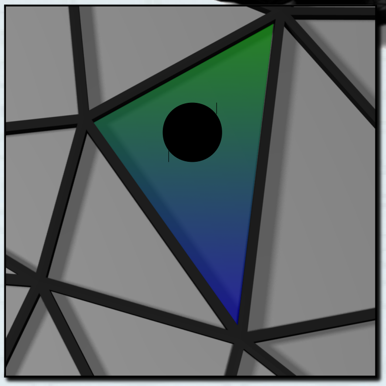


  - PL 2-manifold:
    - Surface triangulation
    - Piecewise linear interpolant
      - Scalar field: valued at 0-simplices + linear interpolation

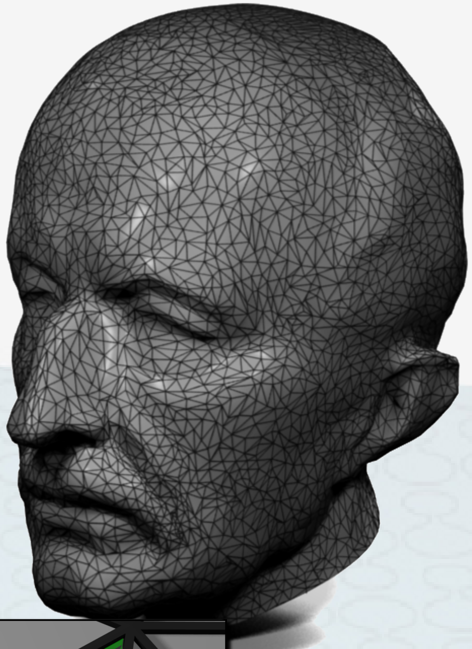
Discrete setting recap (2/2)



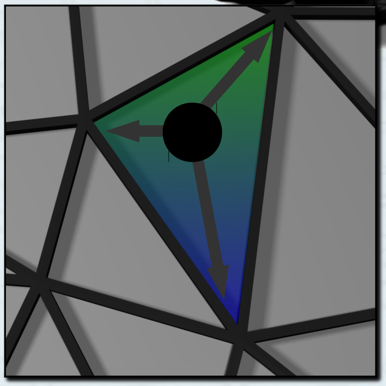
- Triangulation of a manifold $\mathcal{M}$
  - $\mathcal{K}$, such that the union of its simplices is homeomorphic to $\mathcal{M}$
  - PL 2-manifold:
    - Surface triangulation
    - Piecewise linear interpolant
      - Scalar field: valued at 0-simplices + linear interpolation
      - For any point of $\mathcal{K}$, barycentric coordinates



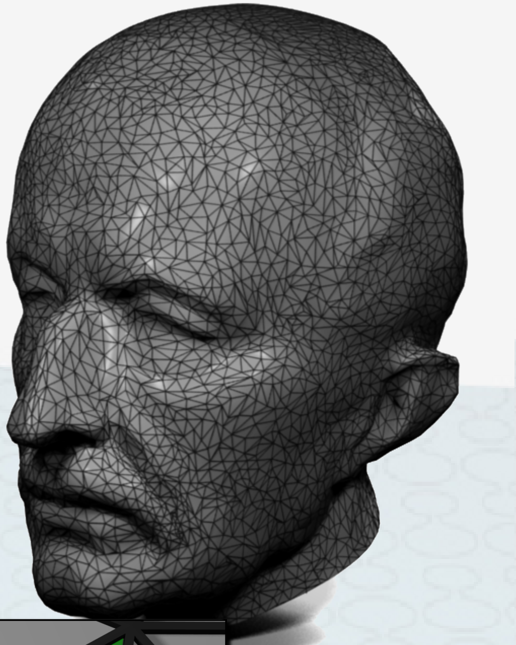
Discrete setting recap (2/2)



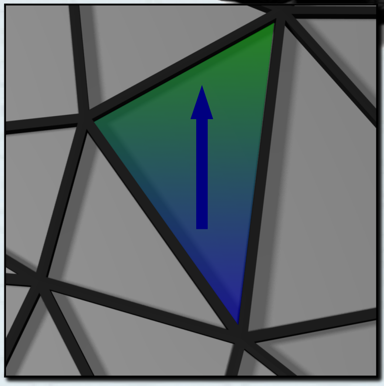
- Triangulation of a manifold $\mathcal{M}$
  - $\mathcal{K}$, such that the union of its simplices is homeomorphic to $\mathcal{M}$
  - PL 2-manifold:
    - Surface triangulation
    - Piecewise linear interpolant
      - Scalar field: valued at 0-simplices + linear interpolation
      - For any point of $\mathcal{K}$, barycentric coordinates



Discrete setting recap (2/2)

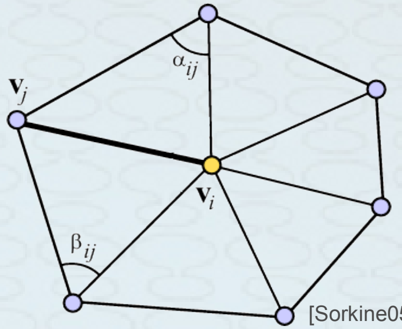


- Triangulation of a manifold $\mathcal{M}$
  - $\mathcal{K}$, such that the union of its simplices is homeomorphic to $\mathcal{M}$
  - PL 2-manifold:
    - Surface triangulation
    - Piecewise linear interpolant
      - Scalar field: valued at 0-simplices + linear interpolation
      - For any point of $\mathcal{K}$, barycentric coordinates
      - Gradient of a scalar field: piecewise constant vector field
      - $\nabla f = \frac{1}{2A} \sum_i f_i (n \times (v_{(i+2)\%3} - v_{(i+1)\%3}))$



Discretization of the Laplace operator

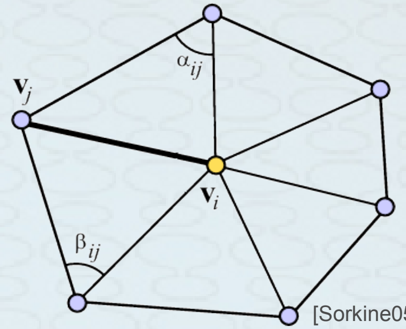
- Discrete Laplace operator**s**
  - Simple interpretation with differential coordinates [Sorkine 2005]
  - $v_i = \delta_i + \frac{1}{d_i}\sum_j v_j$
  - $\delta_i = v_i - \frac{1}{d_i}\sum_j v_j = \frac{1}{d_i}\sum_j (v_i - v_j)$
  - δ_i can be viewed as a discretization of the Laplace-Beltrami operator



[Sorkine05]

Discretization of the Laplace operator

- Discrete Laplace operator**s**
  - Simple interpretation with differential coordinates [Sorkine 2005]
  - $v_i = \delta_i + \frac{1}{d_i}\sum_j v_j$
  - $\delta_i = v_i - \frac{1}{d_i}\sum_j v_j = \frac{1}{d_i}\sum_j (v_i - v_j)$
  - δ_i can be viewed as a discretization of the Laplace-Beltrami operator



- Matrix form
  - $L = I - D^{-1}A, \quad D_{ii} = d_i$, symmetric version L_s :
    - $(L_s)_{ij} = d_i \ (i = j), \quad (L_s)_{ij} = -1 \ (i,j) \in E, \quad (L_s)_{ij} = 0$
  - L_s : graph Laplacian
  - [Pinkall and Polthier 1993]
    - $\delta_i = \frac{1}{\Omega_i}\sum_j \frac{1}{2}(cot\alpha_{ij} + cot\beta_{ij})(v_i - v_j)$
  - … no free lunch :([Wardetzky et al. 2007]

Solving a Laplace equation

- Compute the scalar field f, such that:
  - $\Delta f = 0$
  - $f(v_{c_i}) = f_{c_i}$

Solving a Laplace equation

- Compute the scalar field f, such that:
  - $\Delta f = 0$
  - $f(v_{c_i}) = f_{c_i}$
- Constraint handling [Xu et al. 2009]
  - Many techniques exist (direct elimination, substitution)
  - Penalty method
    - $f^* = argmin(||Lf||^2 + \alpha \sum_{c_i} |f(v_{c_i} - f_{c_i})|)$

Solving a Laplace equation

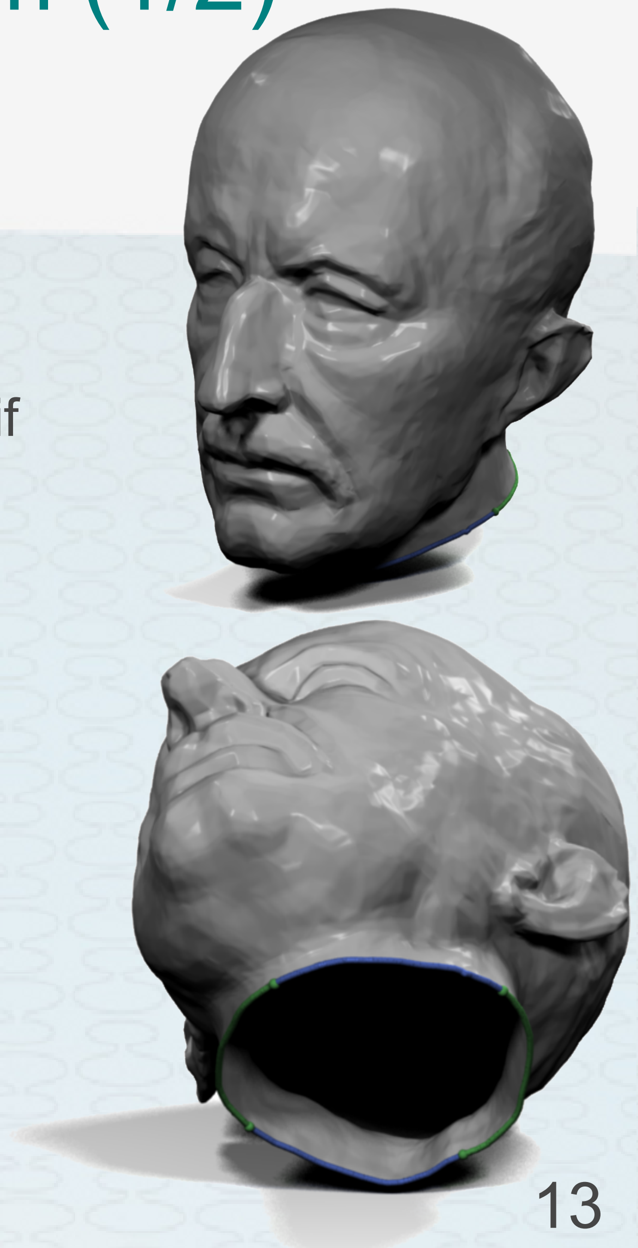
- Compute the scalar field f, such that:
  - $\Delta f = 0$
  - $f(v_{c_i}) = f_{c_i}$
- Constraint handling [Xu et al. 2009]
  - Many techniques exist (direct elimination, substitution)
  - Penalty method
    - $f^* = argmin(||Lf||^2 + \alpha \sum_{c_i} |f(v_{c_i} - f_{c_i})|)$
      $= argmin(||(L+P)f - PC||^2)$
    - $P_{ij} = \alpha \quad (i = j), i \in C_i$

Solving a Laplace equation

- Compute the scalar field f, such that:
  - $\Delta f = 0$
  - $f(v_{c_i}) = f_{c_i}$
- Constraint handling [Xu et al. 2009]
  - Many techniques exist (direct elimination, substitution)
  - Penalty method
    - $$f^* = argmin(||Lf||^2 + \alpha \sum_{c_i} |f(v_{c_i} - f_{c_i})|) = argmin(||(L+P)f - PC||^2)$$
    - $P_{ij} = \alpha \quad (i = j), i \in C_i$
    - Least square problem $||Ax - b||^2$ with $A = L + P, \quad b = PC$
    - Unique solution $f^* = (A^T A)^{-1} A^T b$
  - CHOLMOD library (support for fast updates)

A simple unfolding algorithm (1/2)

- [Floater and Hormann 2005]
  - Given a surface $\mathcal{S}$ with disc topology
  - A harmonic map $\phi : \mathcal{S} \to \mathcal{D}$ is one to one if

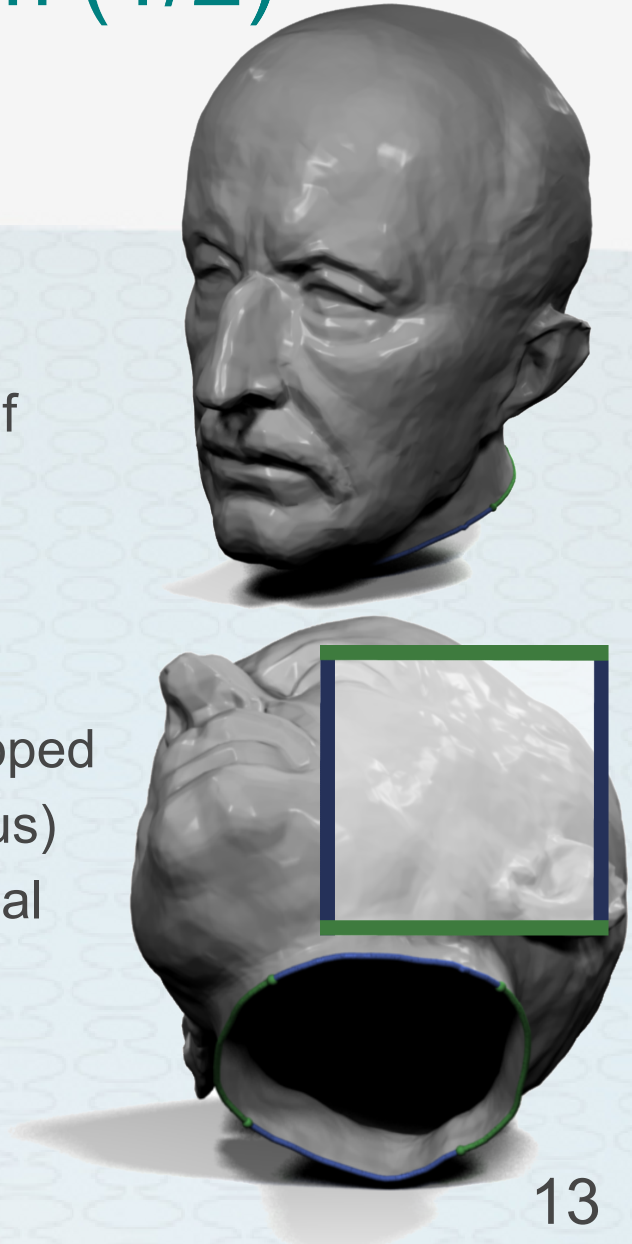


A simple unfolding algorithm (1/2)

- [Floater and Hormann 2005]
  - Given a surface $\mathcal{S}$ with disc topology
  - A harmonic map $\phi : \mathcal{S} \to \mathcal{D}$ is one to one if
    - ϕ maps homeomorphically $\partial\mathcal{S}$ to $\partial\mathcal{D}$ and $\mathcal{D}$ is a convex region of $\mathbb{R}^2$

A simple unfolding algorithm (1/2)

- [Floater and Hormann 2005]
  - Given a surface $\mathcal{S}$ with disc topology
  - A harmonic map $\phi : \mathcal{S} \rightarrow \mathcal{D}$ is one to one if
    - ϕ maps homeomorphically $\partial\mathcal{S}$ to $\partial\mathcal{D}$ and $\mathcal{D}$ is a convex region of $\mathbb{R}^2$
- Uniformization theorem
  - Any simply connected surface can be mapped conformally to its canonical domain (Mobius)
  - There exist harmonic maps being conformal
  - Good **heuristic**
    - Boundary arc length parameterization
    - Low distortion boundary mapping



A simple unfolding algorithm (2/2)

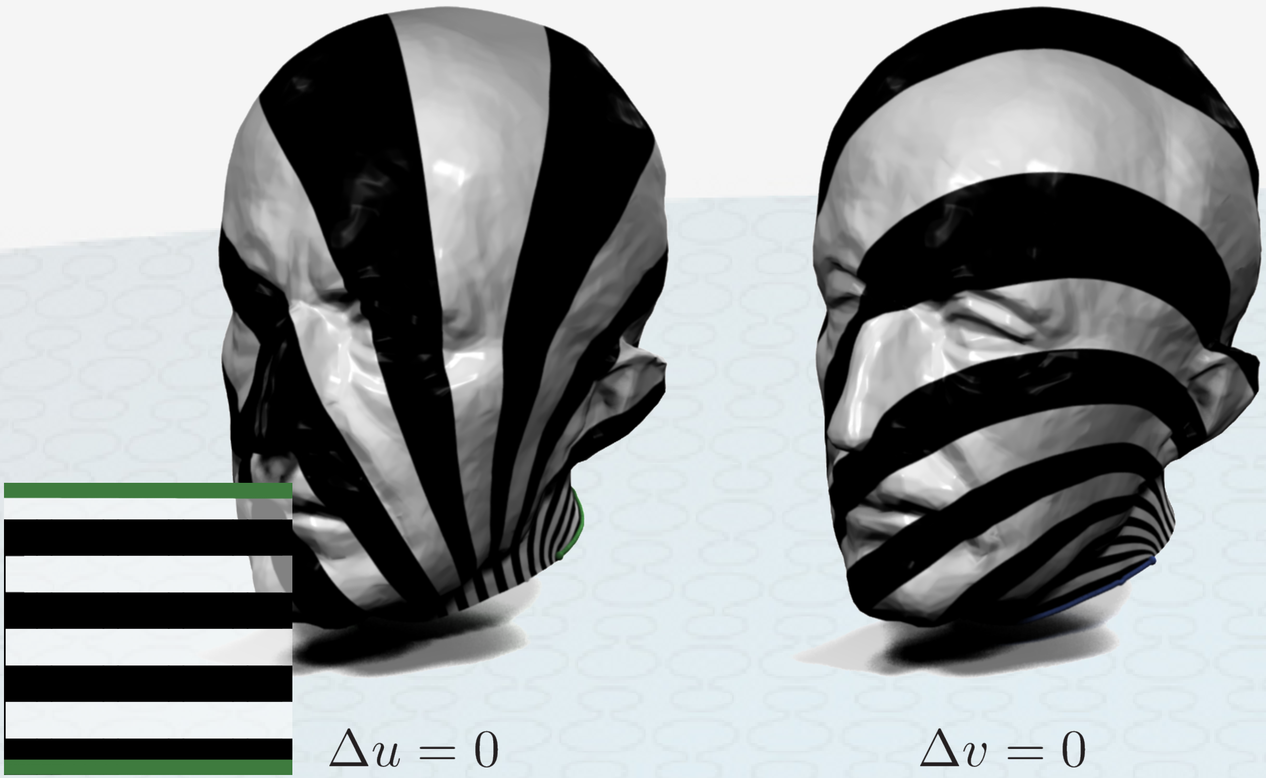
A simple unfolding algorithm (2/2)

$$\Delta u = 0$$

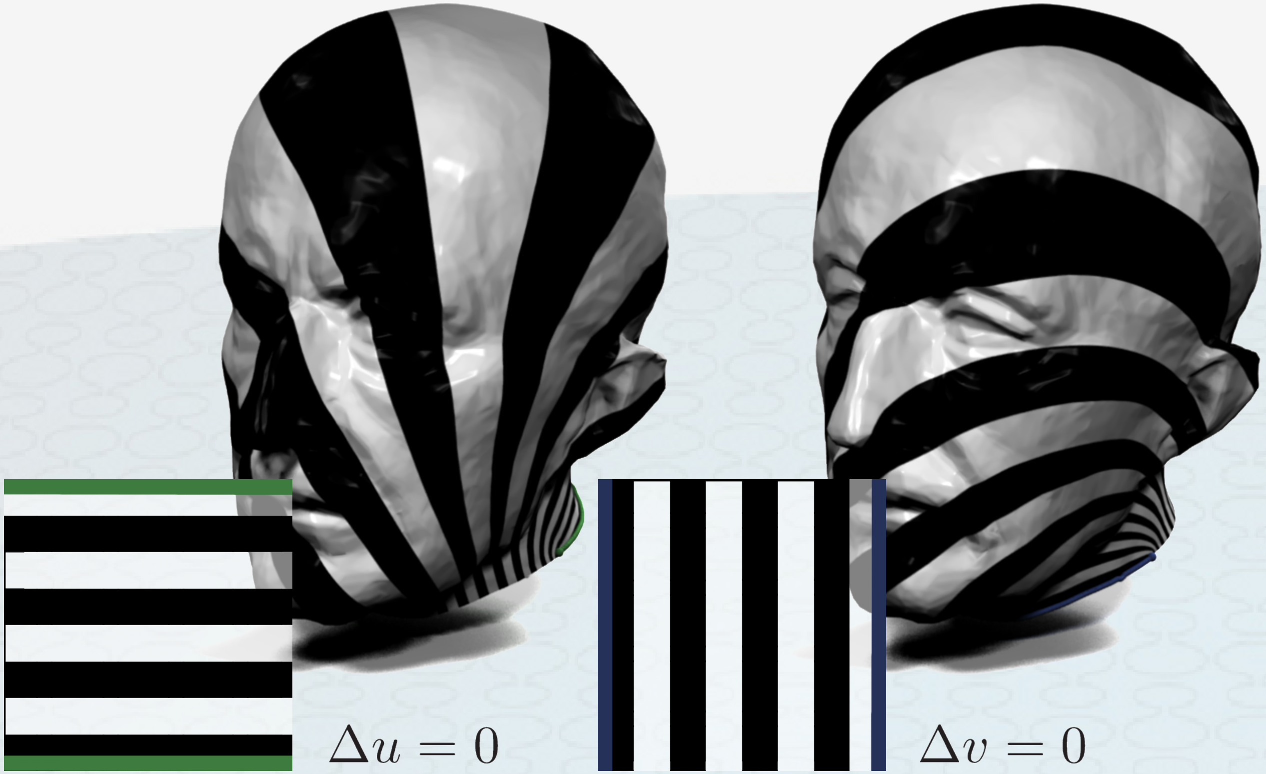
A simple unfolding algorithm (2/2)

$\Delta u = 0$

A simple unfolding algorithm (2/2)



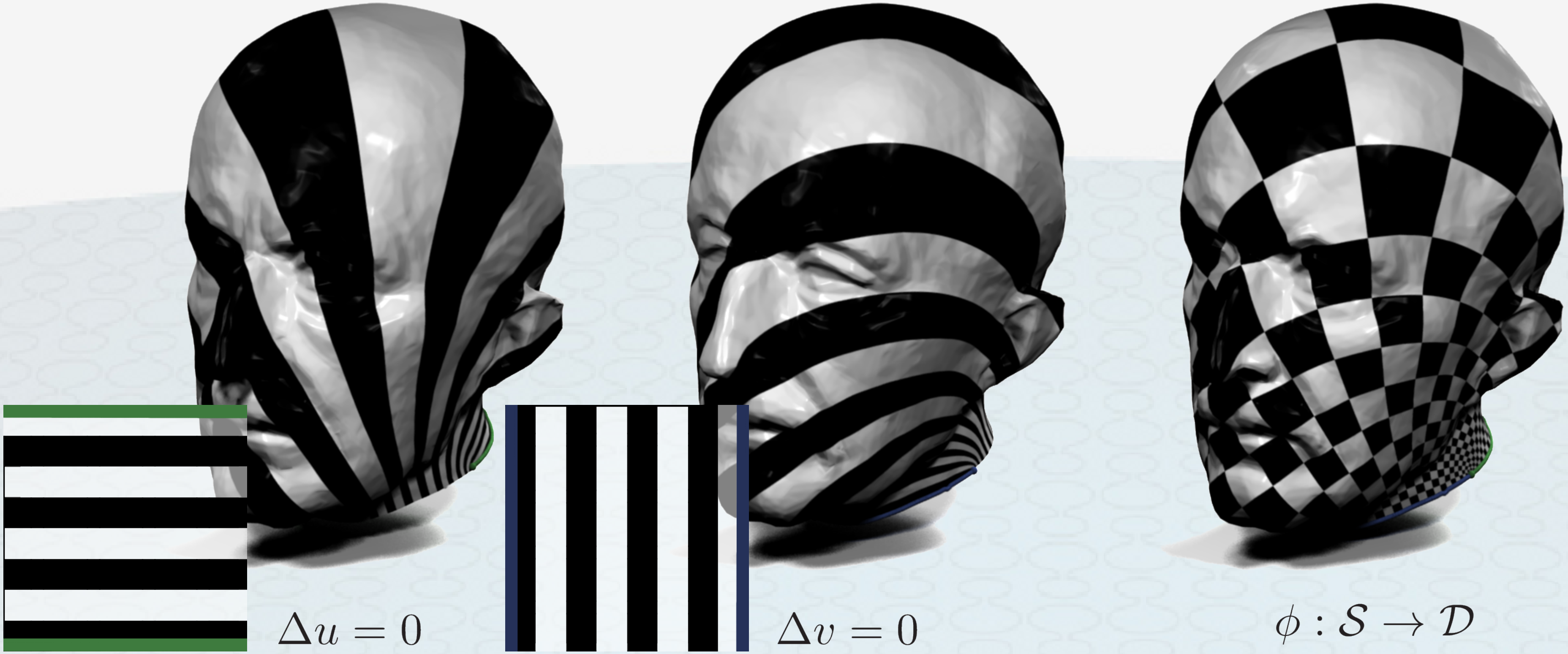
A simple unfolding algorithm (2/2)



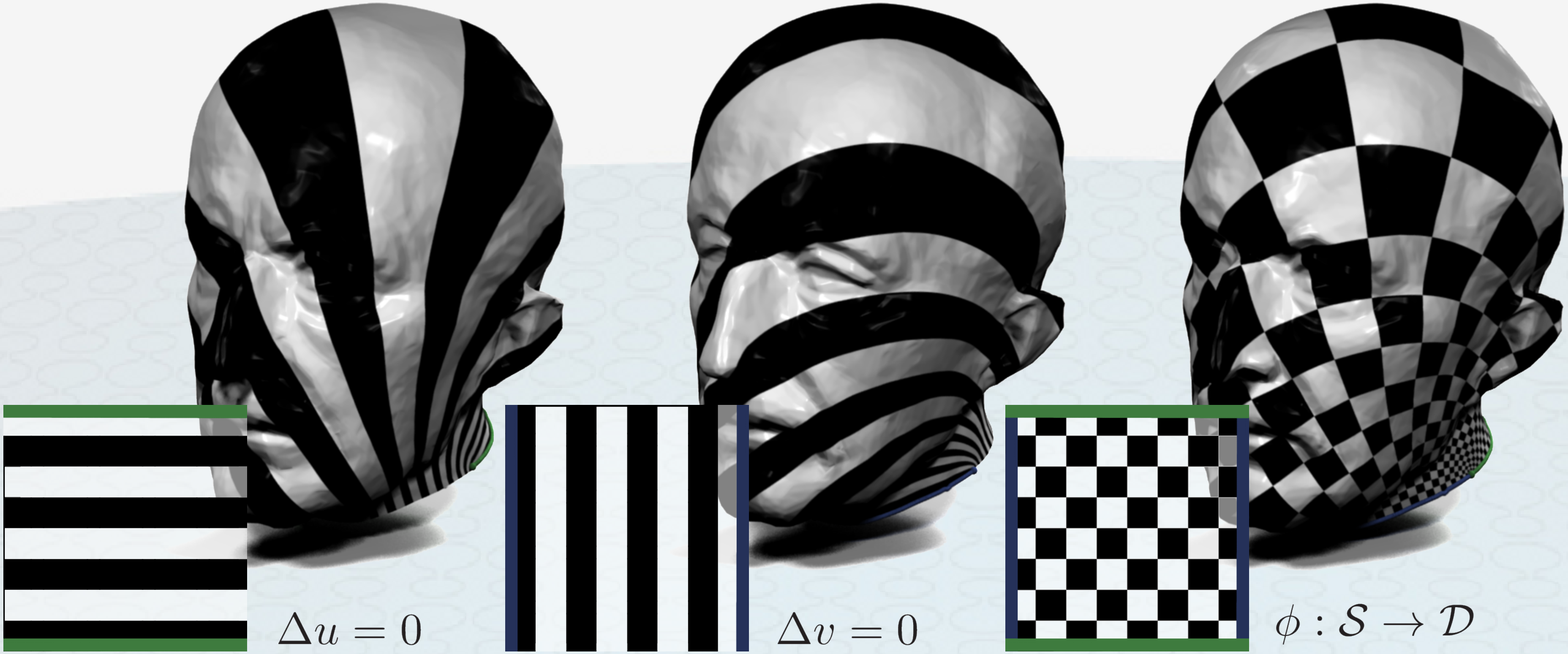
$\Delta u = 0$

$\Delta v = 0$

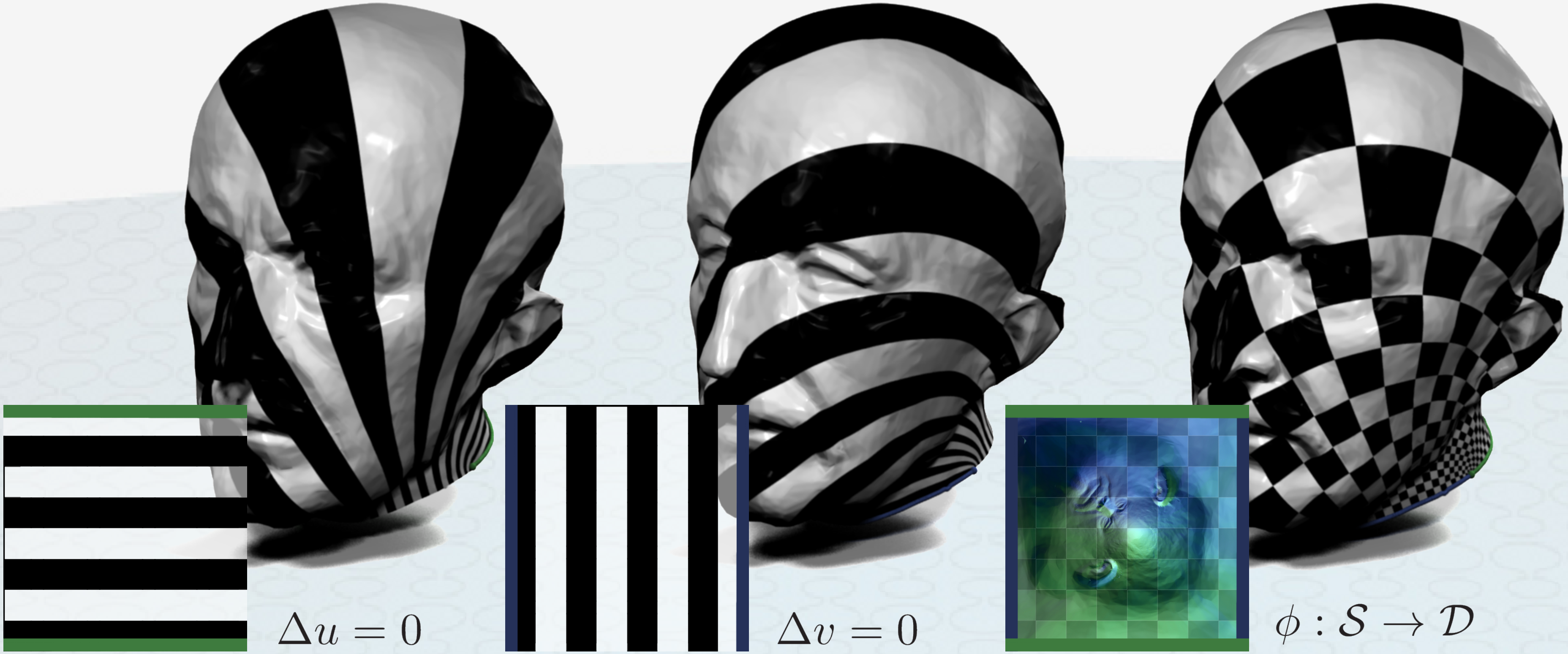
A simple unfolding algorithm (2/2)



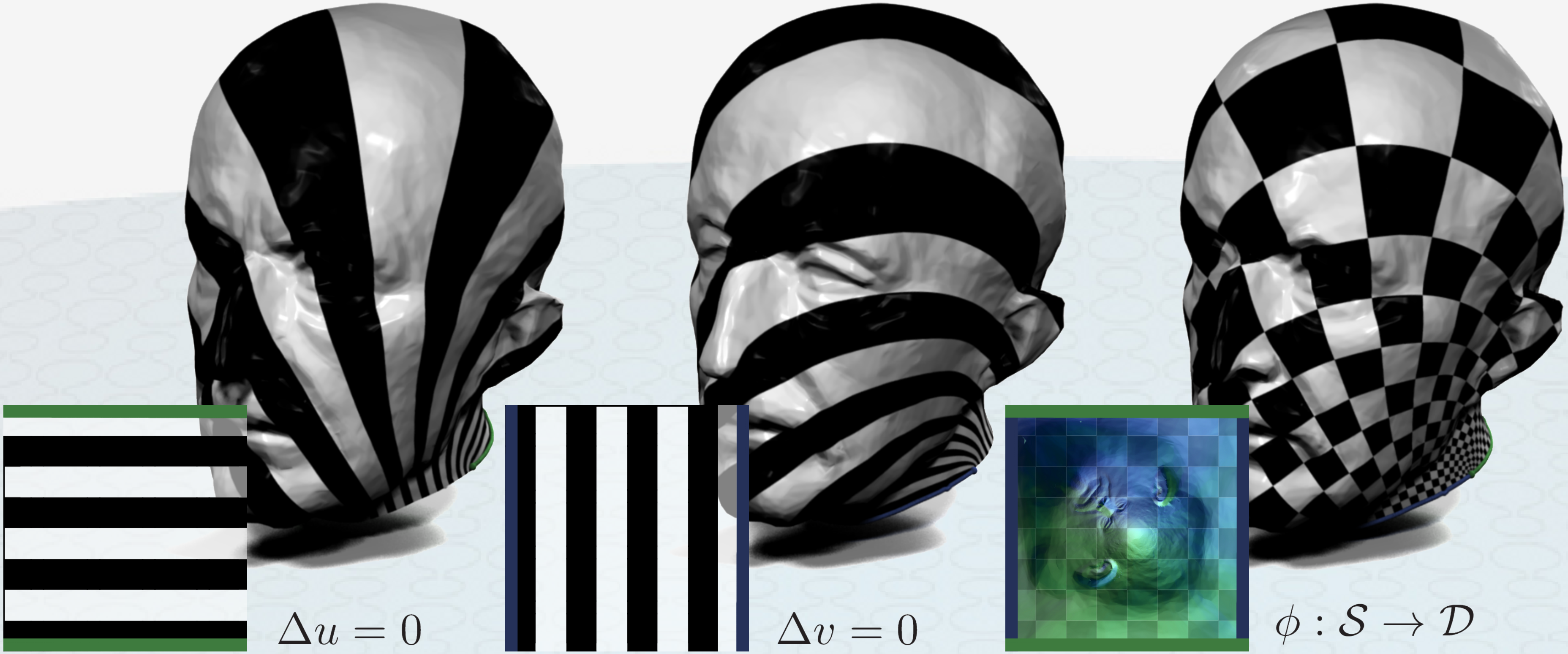
A simple unfolding algorithm (2/2)



A simple unfolding algorithm (2/2)

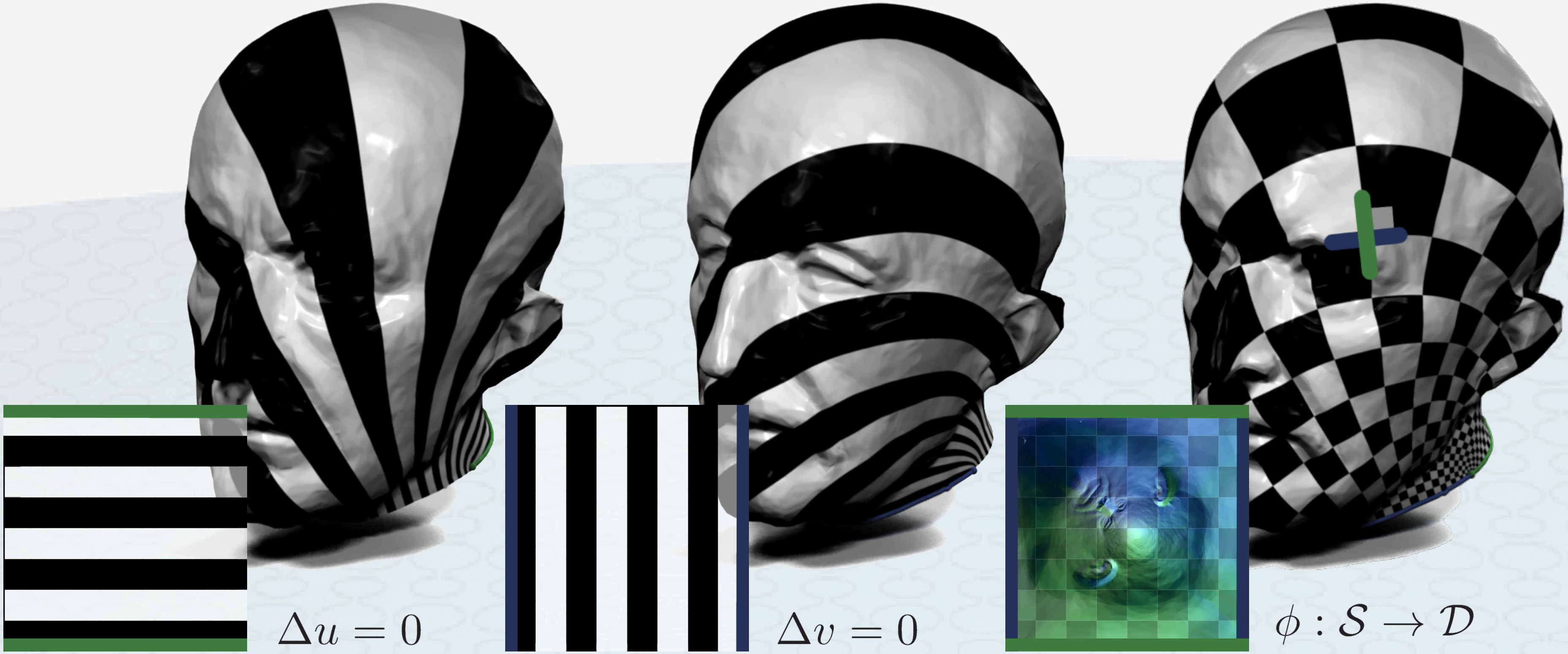


A simple unfolding algorithm (2/2)



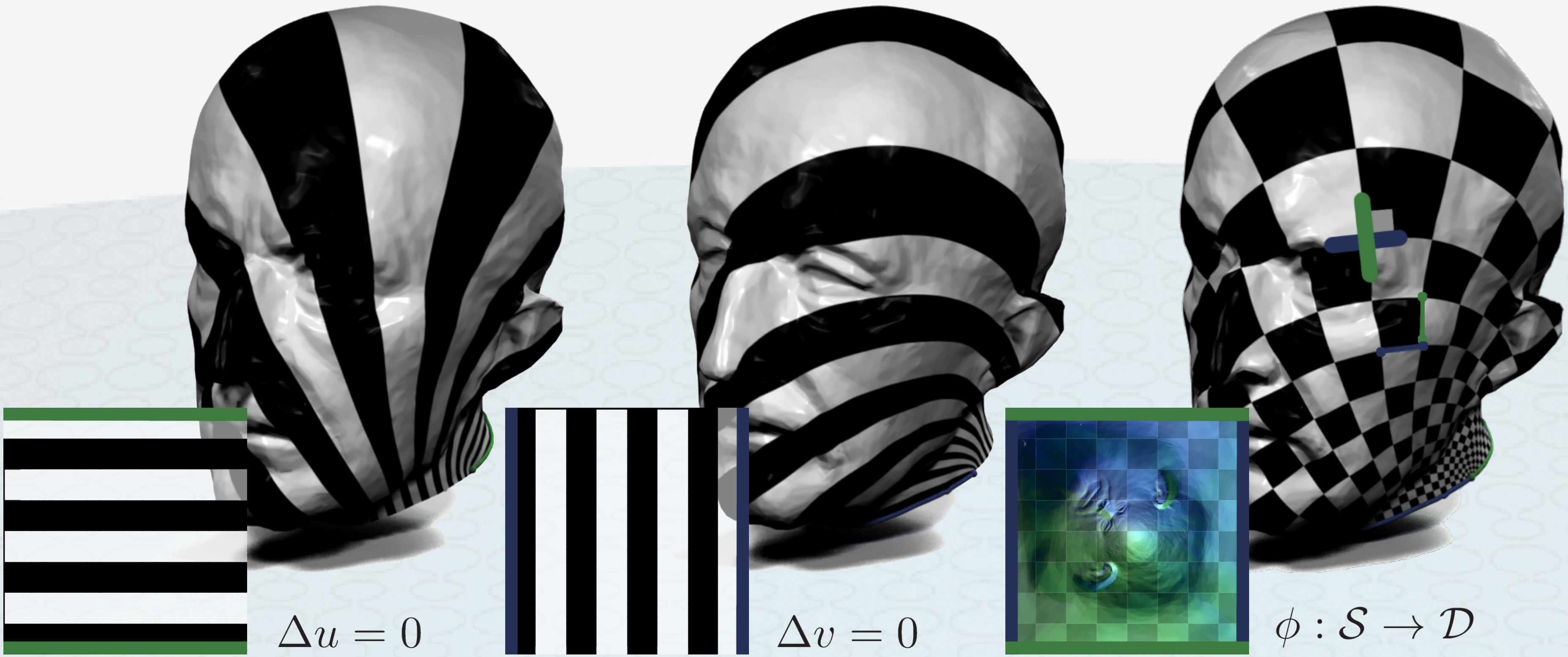
- Simple and fast implementation (~ 200k triangles per second)

A simple unfolding algorithm (2/2)



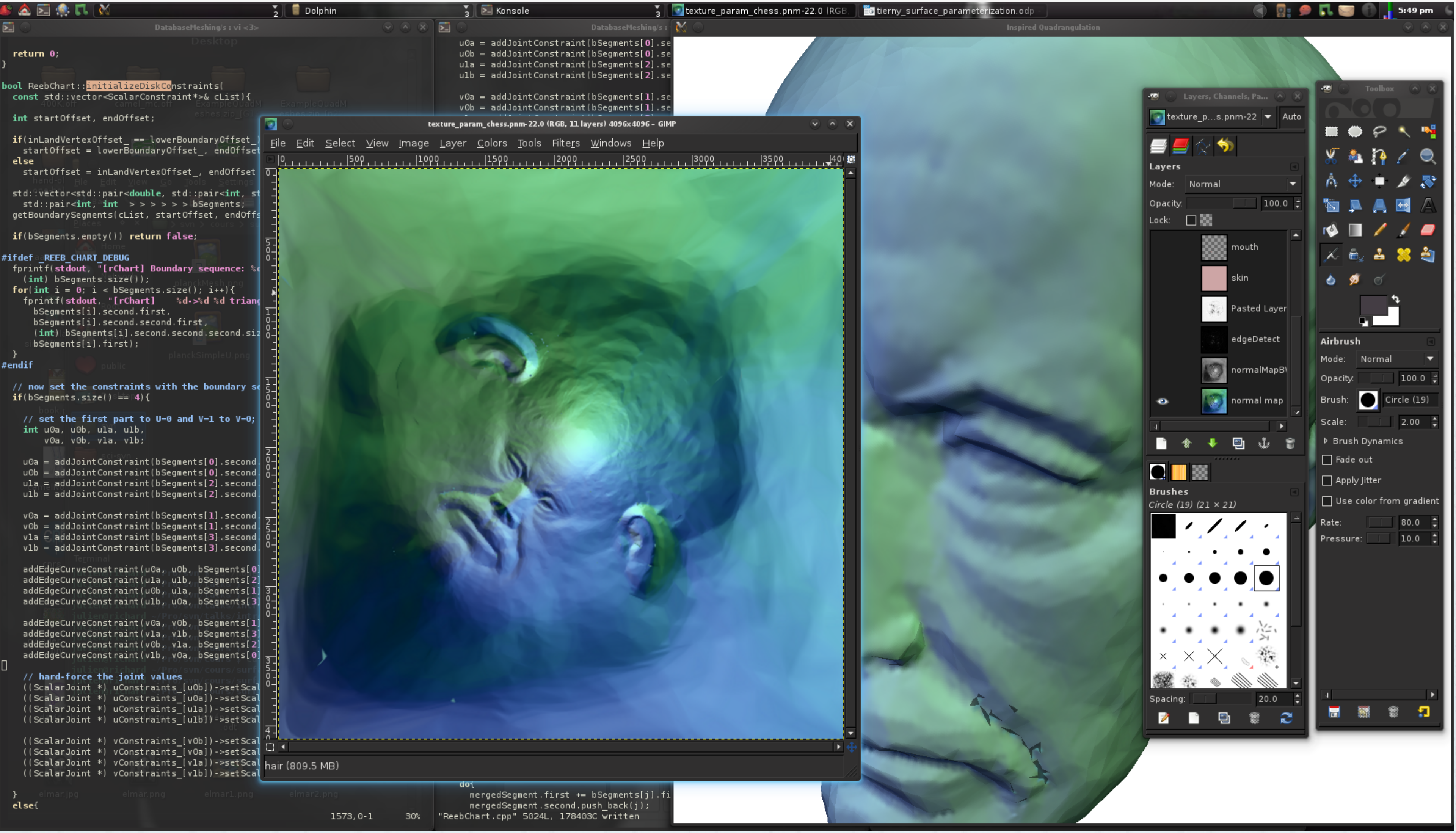
- Simple and fast implementation (~ 200k triangles per second)
- Decent conformal approximation (for low distortion boundary mappings)
  - Orthogonality ($\nabla u \cdot \nabla v = 0$)

A simple unfolding algorithm (2/2)

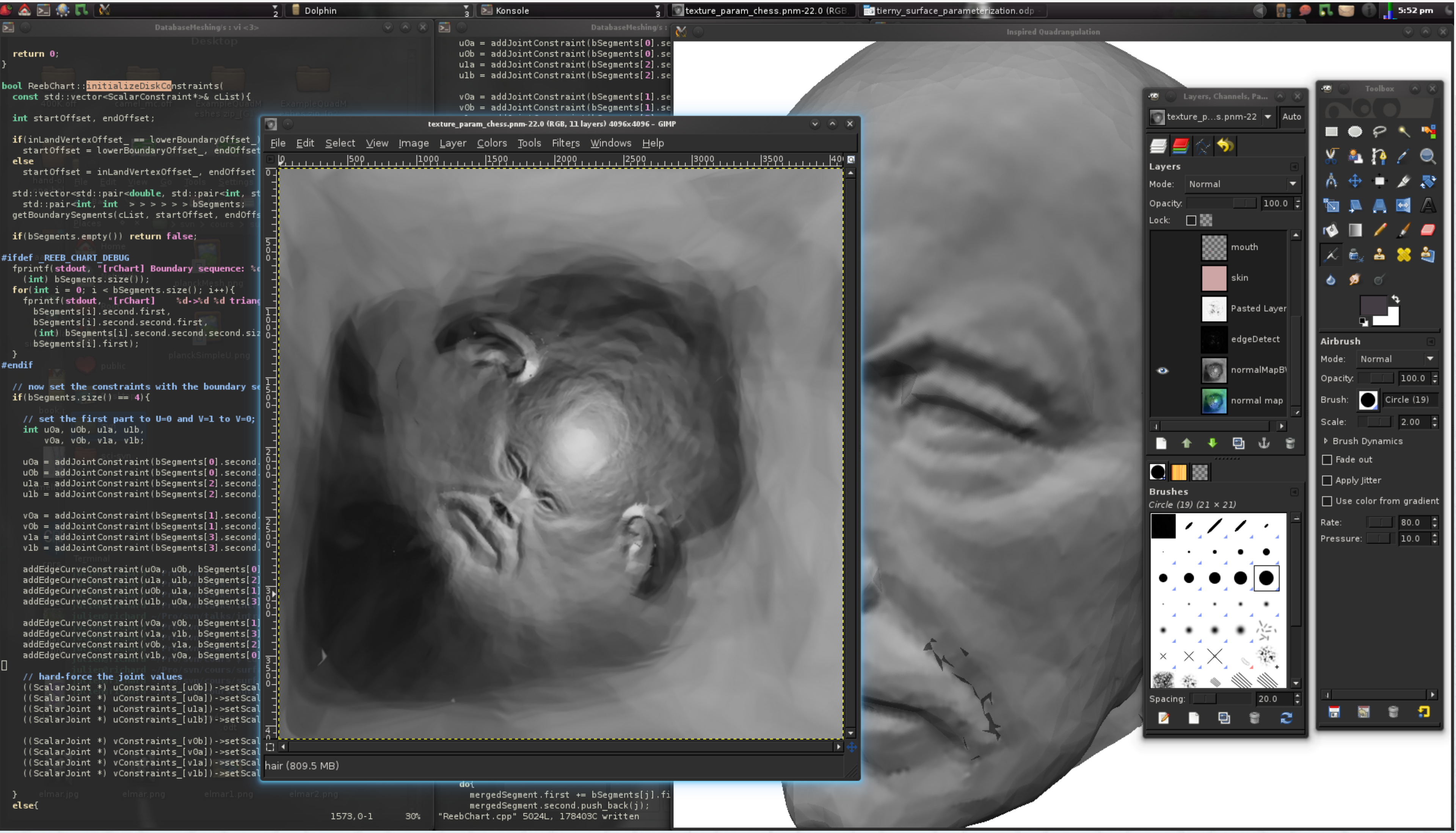


- Simple and fast implementation (~ 200k triangles per second)
- Decent conformal approximation (for low distortion boundary mappings)
  - Orthogonality ($\nabla u \cdot \nabla v = 0$)
  - Local isotropy ($||\nabla u|| = ||\nabla v||$)
- Geometric interpretation: extreme smoothing

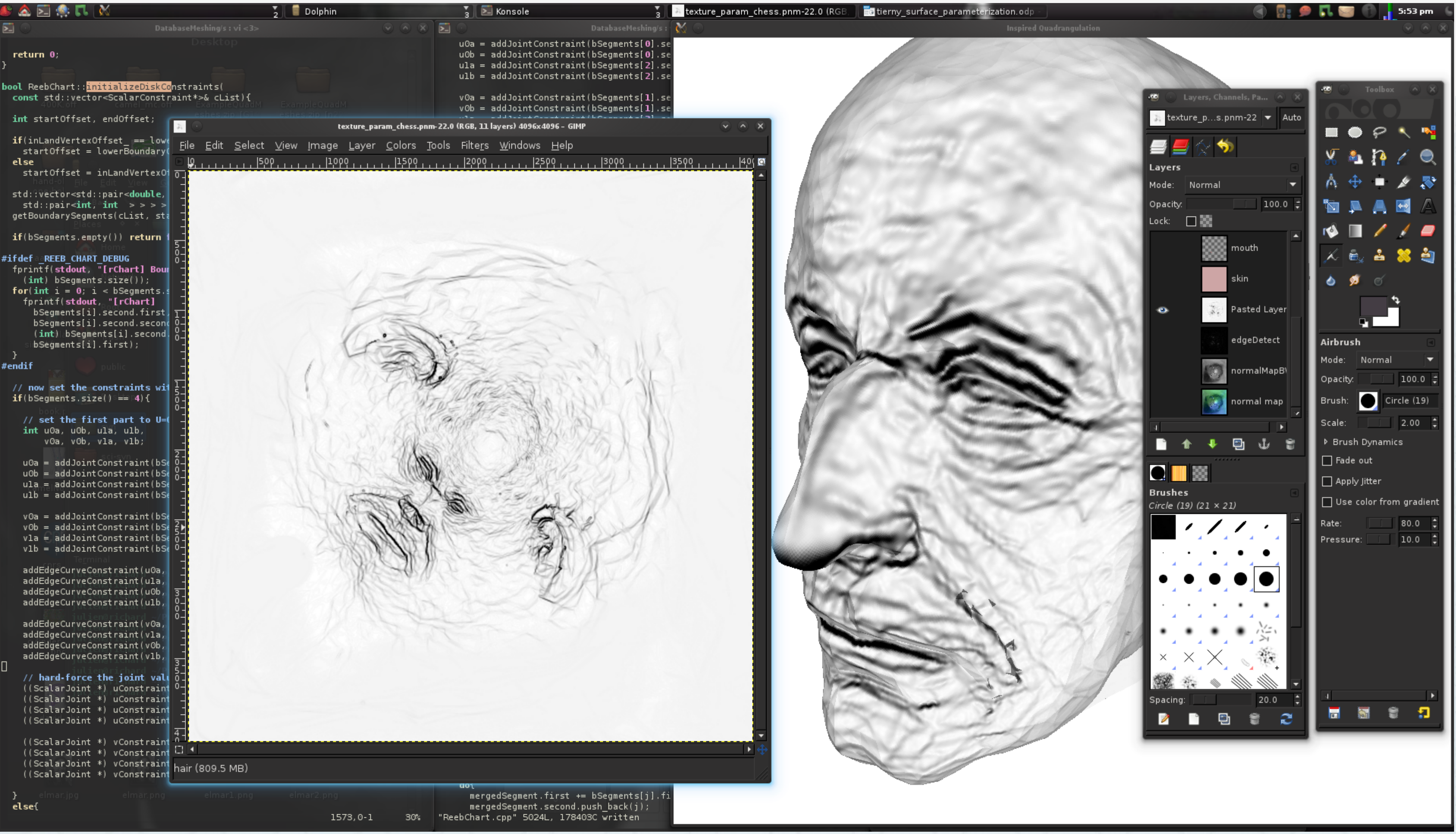
How to (… I'm not an artist)



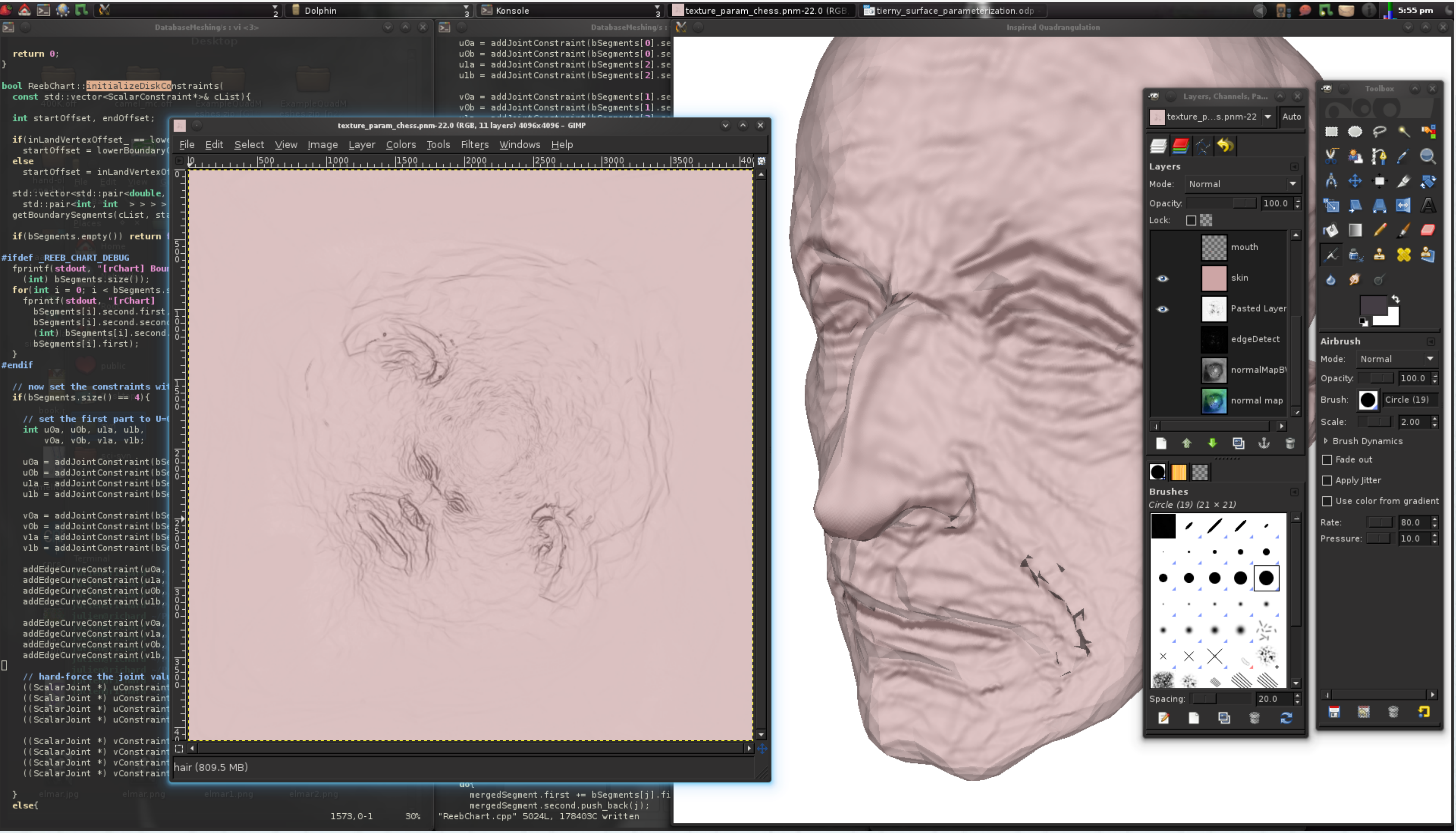
How to (… I'm not an artist)



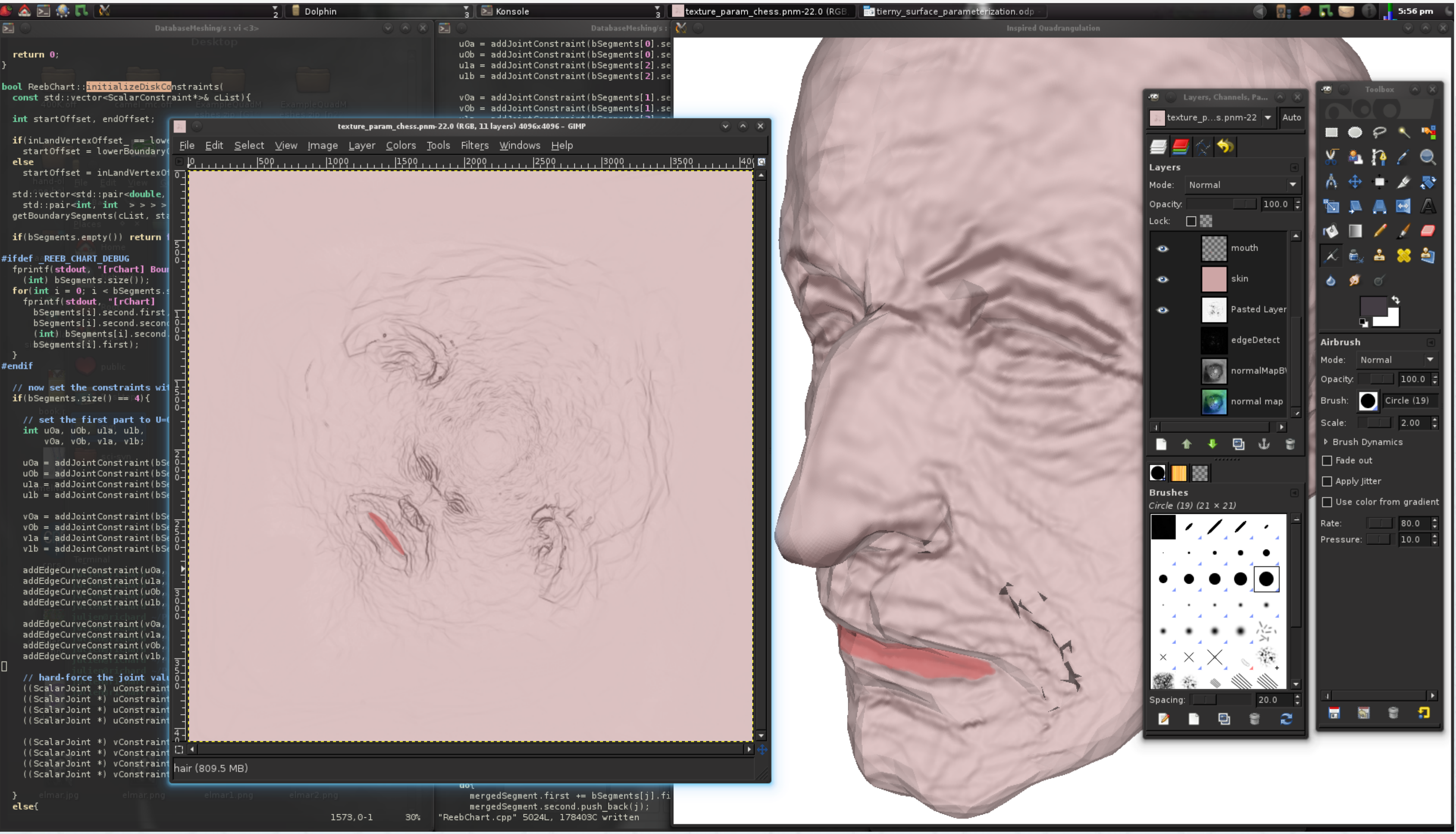
How to (… I'm not an artist)



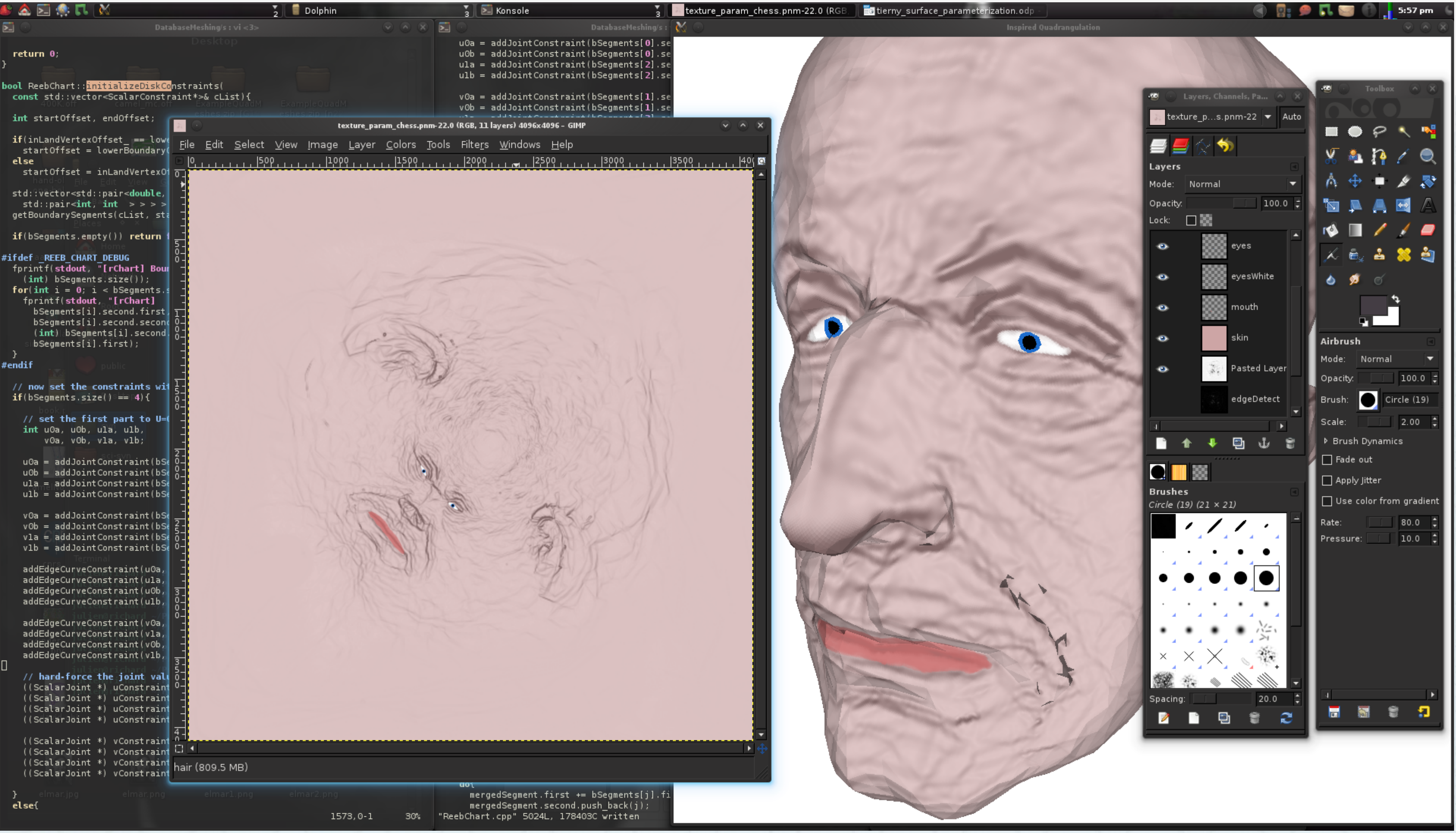
How to (… I'm not an artist)



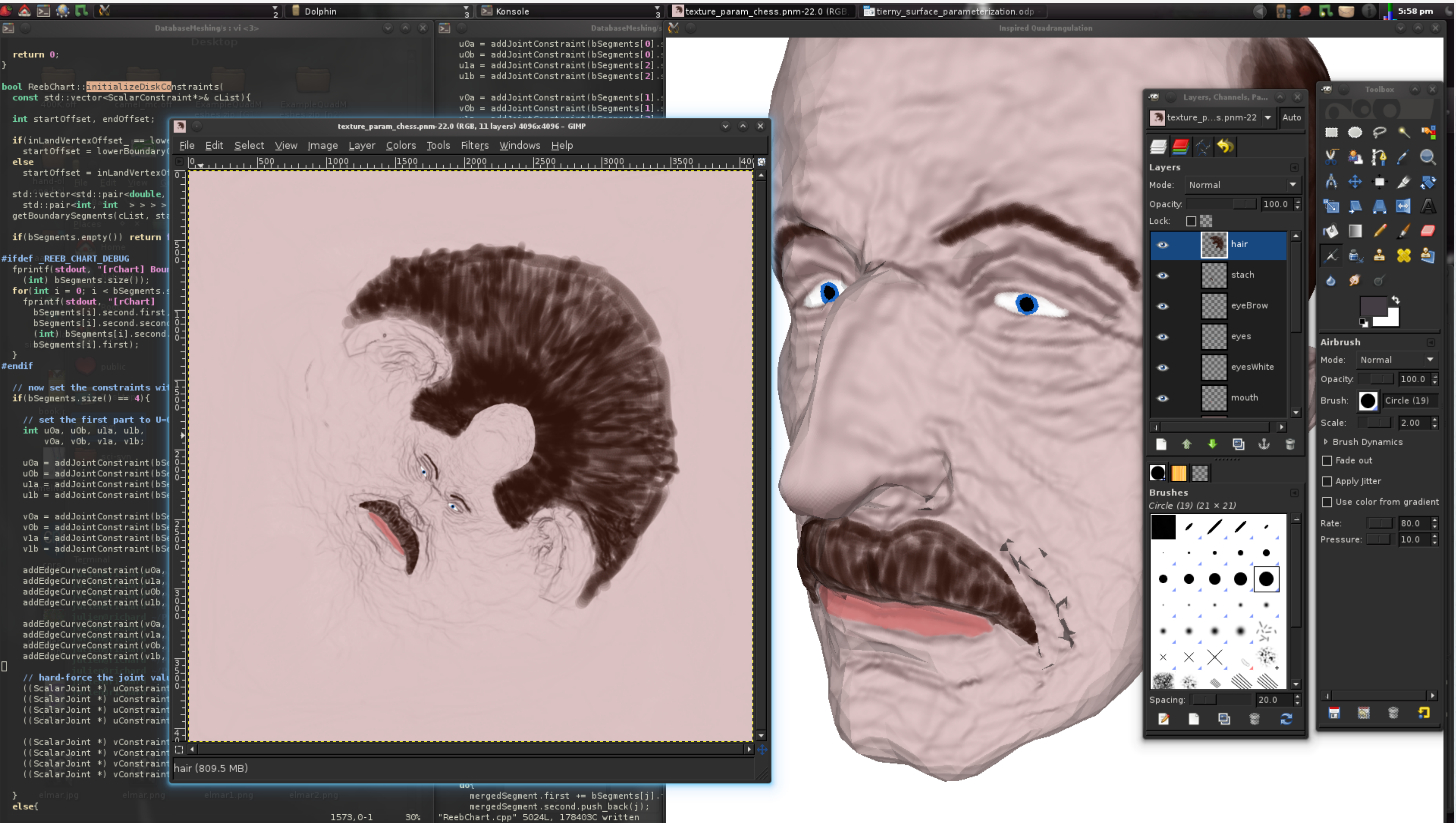
How to (… I'm not an artist)



How to (… I'm not an artist)



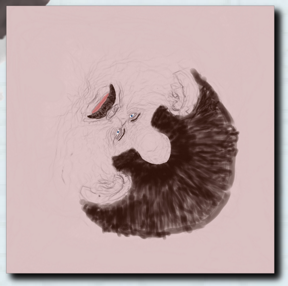
How to (… I'm not an artist)



Et voila!

Et voila!

Plausible
Planck



Et voila!

Plausible Planck

Afro Planck

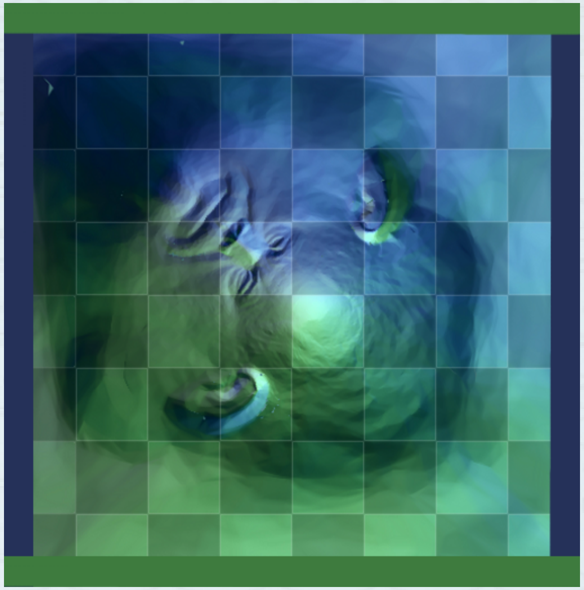
Et voila!

Et voila!

- Fast algorithm, easy to implement, decent results with a good boundary
- You're now able to write your own geometry texturing program

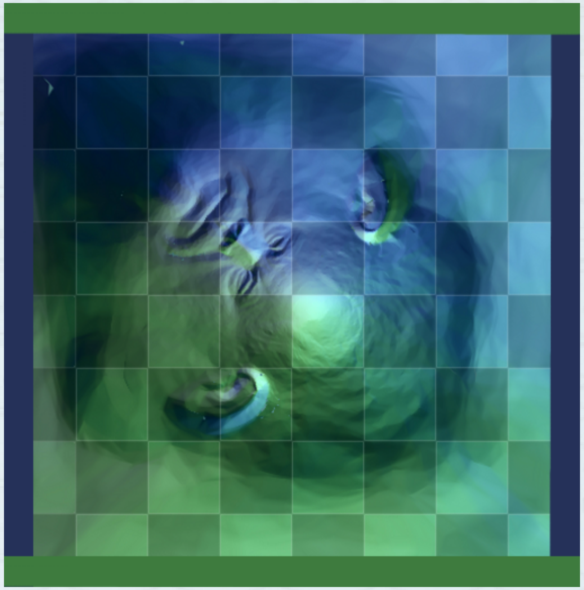
Mission accomplished?

- Well...
  - Constraints on the boundary's shape
  - Convexity of the planar domain

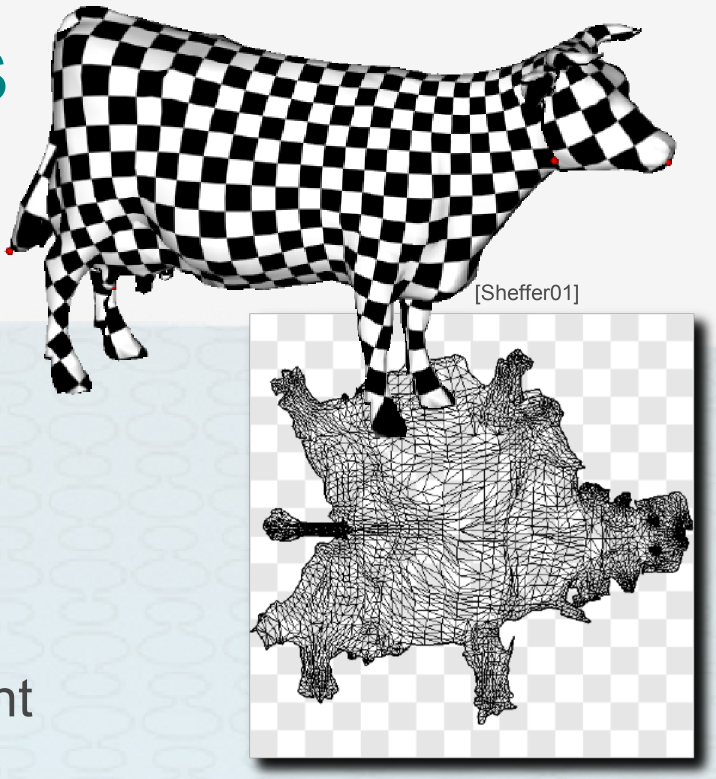


Mission accomplished?

- Well...
  - Constraints on the boundary's shape
  - Convexity of the planar domain
  - Induces important area distortion
  - Significant waste of texture space
- Need for truly conformal parameterizations
- Need for boundary-free algorithms

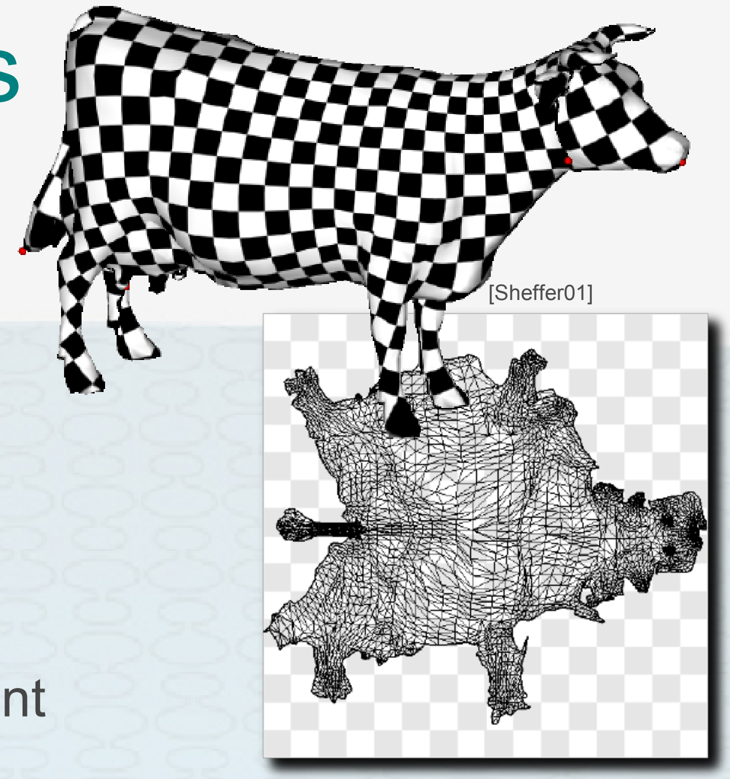


Boundary free algorithms



- Before Least Squares Conformal Maps
  - MIPS [Hormann and Greiner 2000]
  - ABF [Sheffer and de Sturler 2001]
  - Arbitrary cuts, no convexity requirement
  - Iterative solvers (slow convergence)

Boundary free algorithms



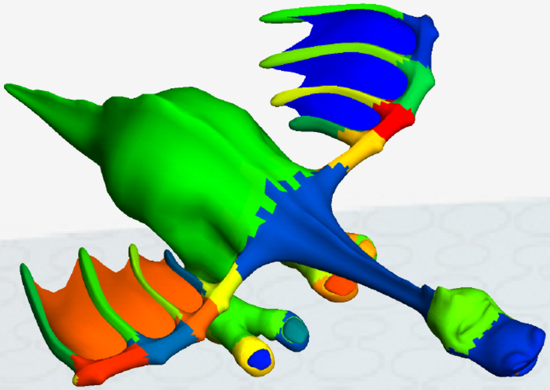
[Sheffer01]

- Before Least Squares Conformal Maps
  - MIPS [Hormann and Greiner 2000]
  - ABF [Sheffer and de Sturler 2001]
  - Arbitrary cuts, no convexity requirement
  - Iterative solvers (slow convergence)
- Least Squares Conformal Maps [Levy et al. 2002]:
  - First linear method
  - Unique solution
  - Few triangle flips in practice
- Set the bar higher :)

Least Squares Conformal Maps

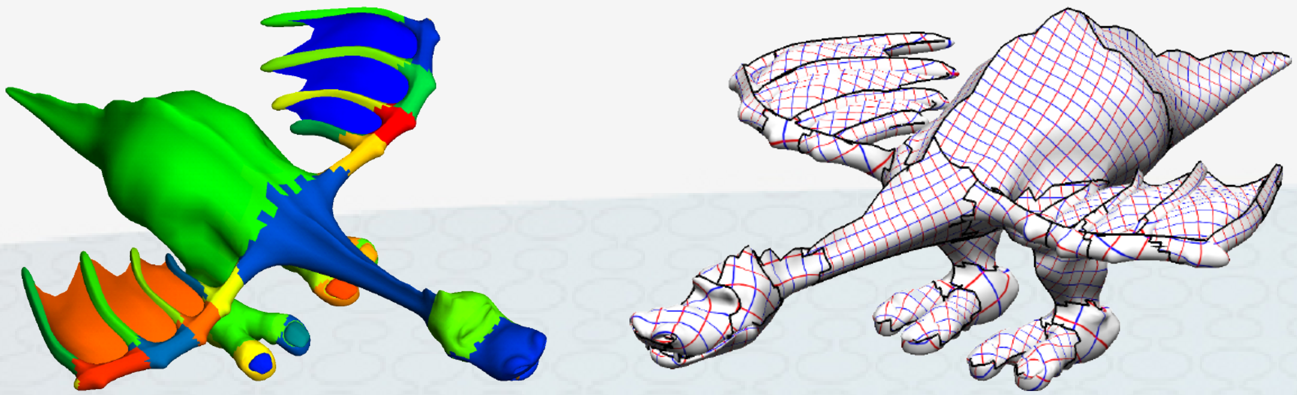
- A complete texturing framework
  - Automatic “atlas” generation
  - **Fast boundary free conformal parameterization** (Blender, Silo)
  - Texture packing (UVatlas of DirectX)

Least Squares Conformal Maps



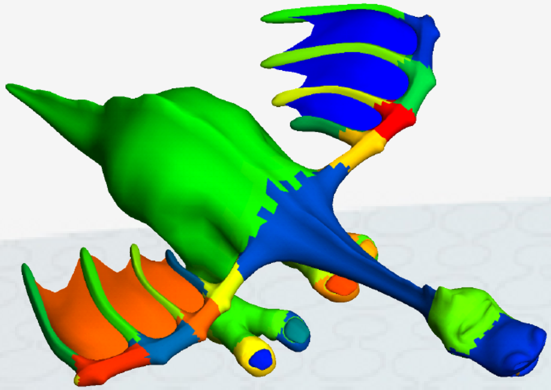
- A complete texturing framework
  - Automatic “atlas” generation
  - **Fast boundary free conformal parameterization** (Blender, Silo)
  - Texture packing (UVatlas of DirectX)

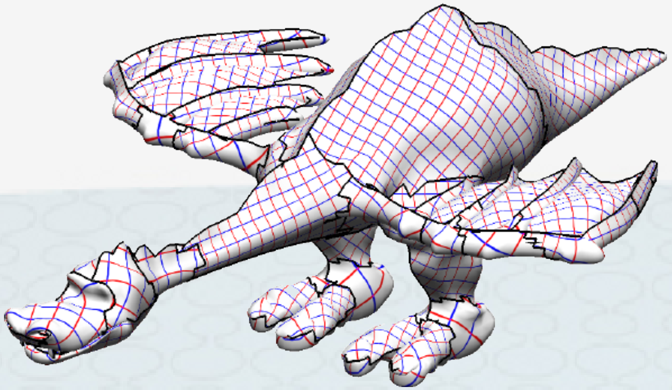
Least Squares Conformal Maps

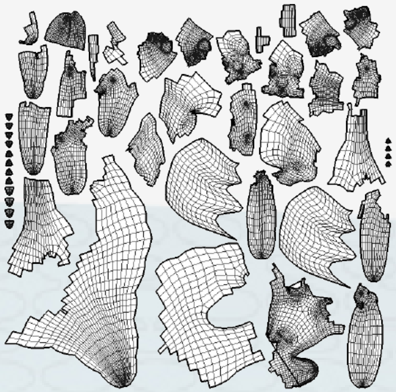


- A complete texturing framework
  - Automatic “atlas” generation
  - **Fast boundary free conformal parameterization** (Blender, Silo)
  - Texture packing (UVatlas of DirectX)

Least Squares Conformal Maps

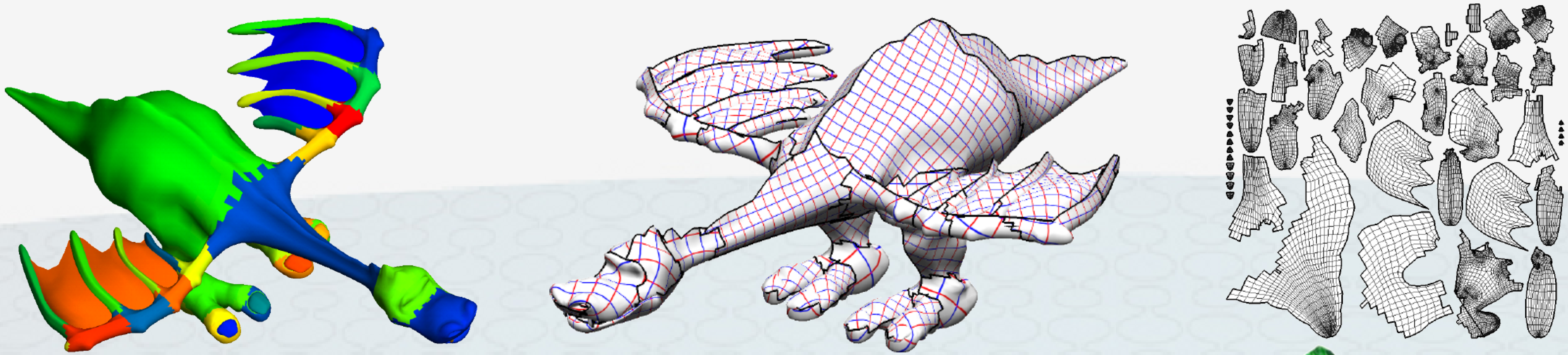






- A complete texturing framework
  - Automatic “atlas” generation
  - **Fast boundary free conformal parameterization** (Blender, Silo)
  - Texture packing (UVatlas of DirectX)

Least Squares Conformal Maps

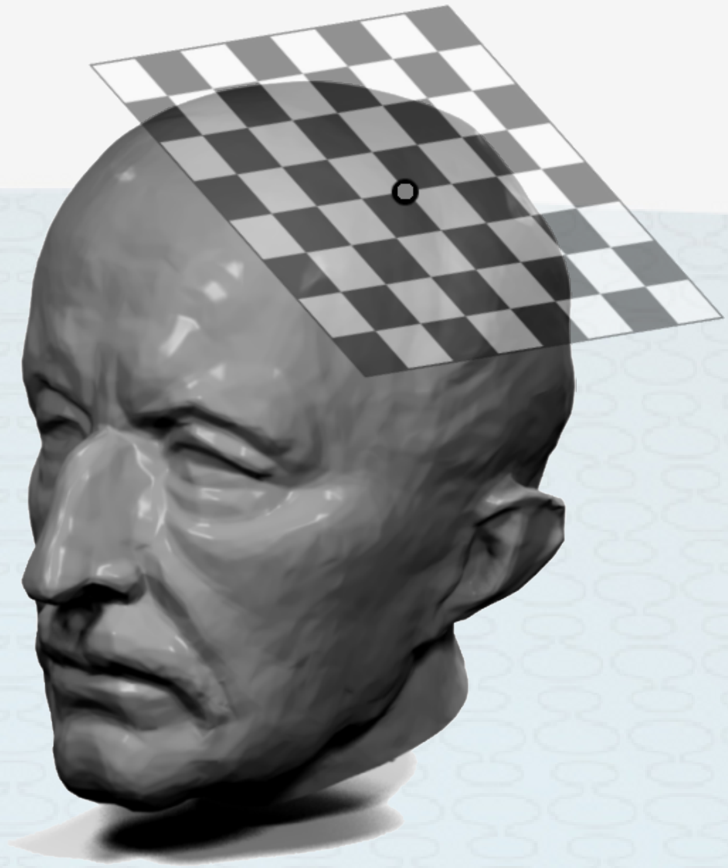


[Levy02]

- A complete texturing framework
  - Automatic “atlas” generation
  - **Fast boundary free conformal parameterization** (Blender, Silo)
  - Texture packing (UVatlas of DirectX)

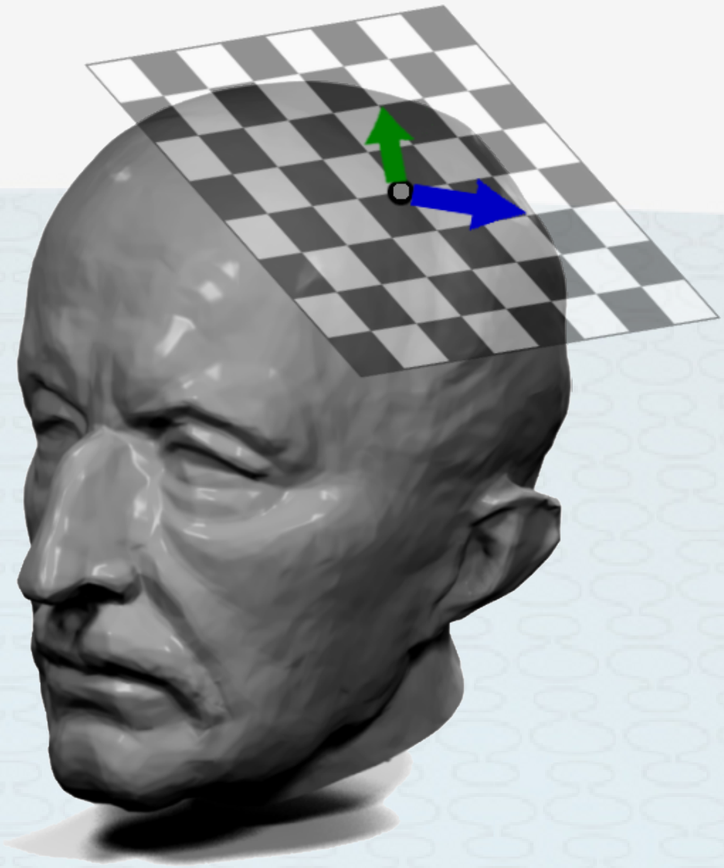
Back to the roots

- Key idea
  - Penalize the violation of the Cauchy Riemann eq. (Least Squares sense)



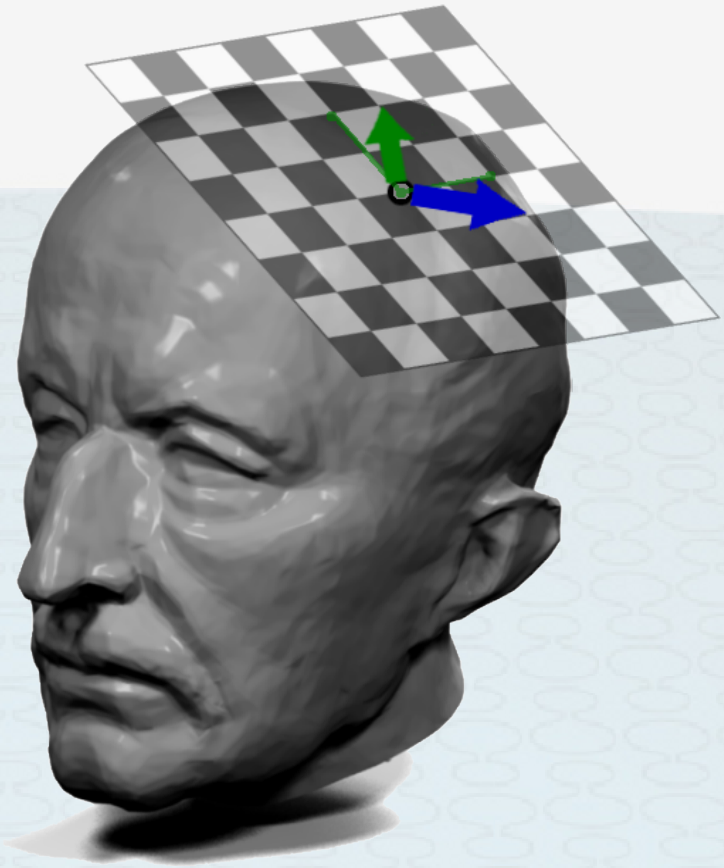
Back to the roots

- Key idea
  - Penalize the violation of the Cauchy Riemann eq. (Least Squares sense)
- Back to the definition of a conformal map
  - For each triangle, define consistently an orthonormal basis (x, y)



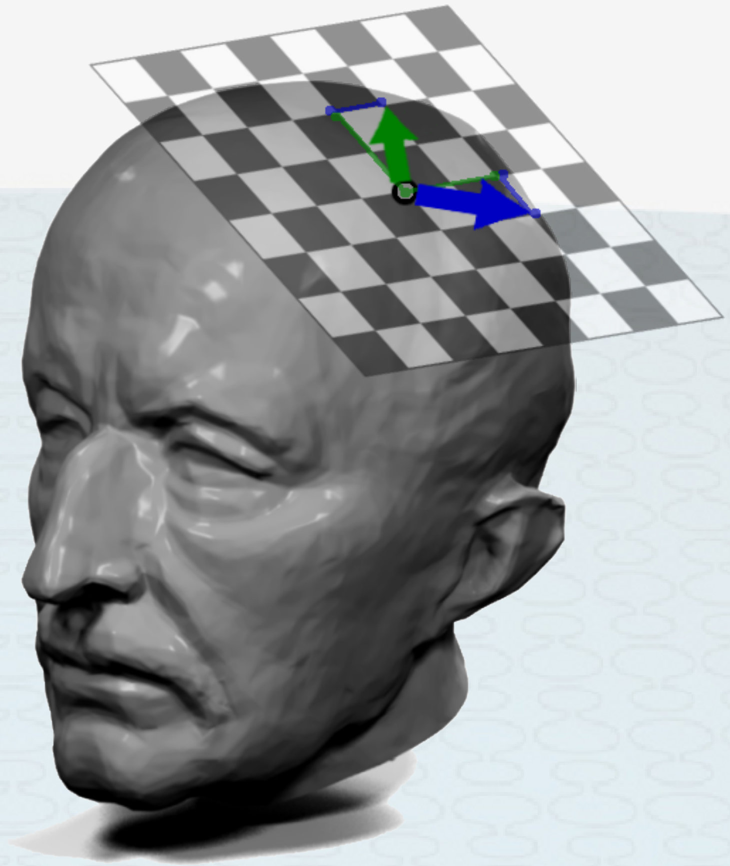
Back to the roots

- Key idea
  - Penalize the violation of the Cauchy Riemann eq. (Least Squares sense)
- Back to the definition of a conformal map
  - For each triangle, define consistently an orthonormal basis (x, y)
  - $\dfrac{\partial u}{\partial x} = \dfrac{\partial v}{\partial y}$



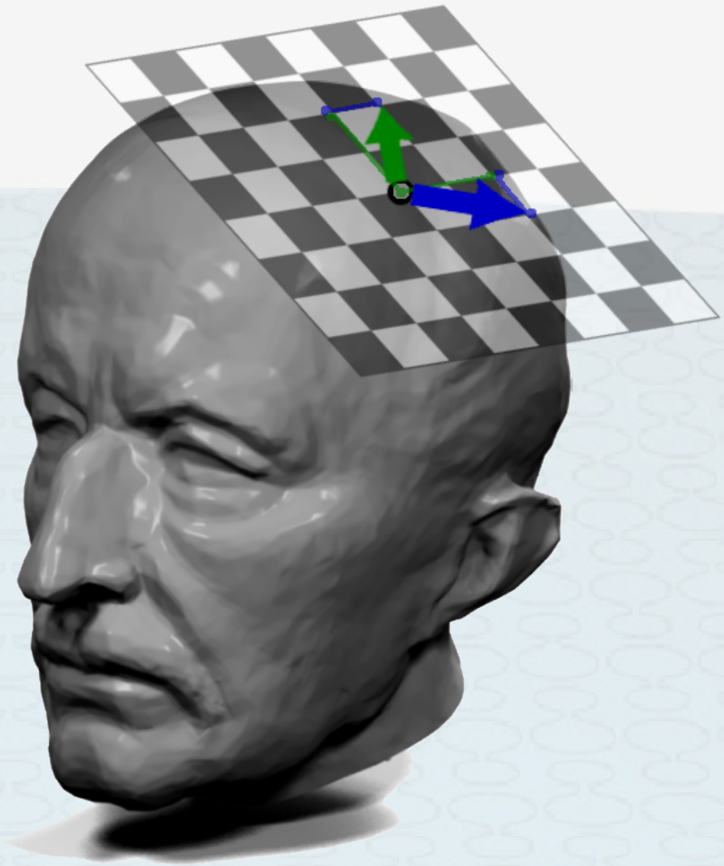
Back to the roots

- Key idea
  - Penalize the violation of the Cauchy Riemann eq. (Least Squares sense)
- Back to the definition of a conformal map
  - For each triangle, define consistently an orthonormal basis (x, y)
  - $\frac{\partial u}{\partial x} = \frac{\partial v}{\partial y} \qquad \frac{\partial u}{\partial y} = -\frac{\partial v}{\partial x}$

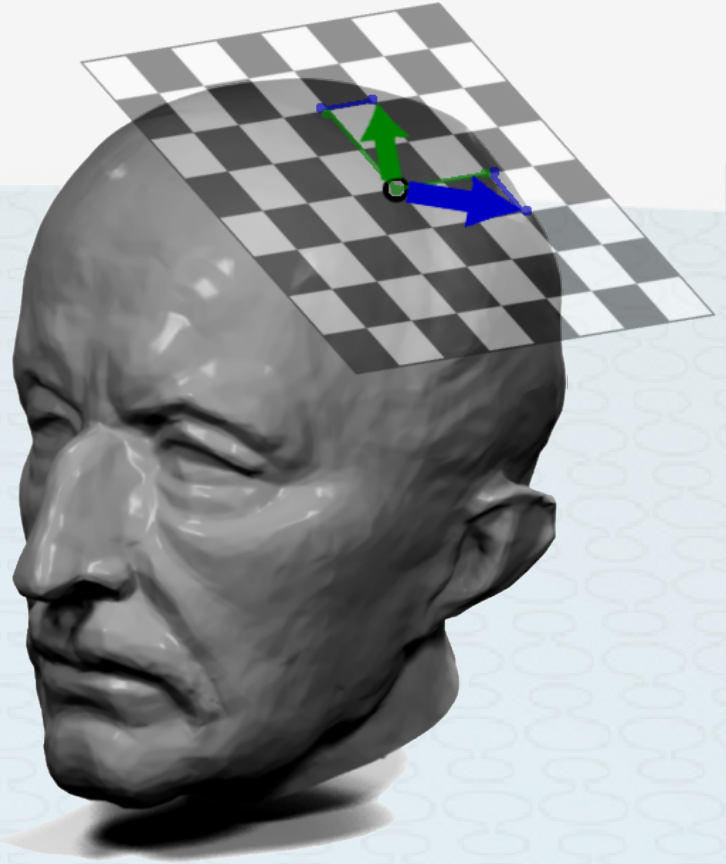


Back to the roots

- Key idea
  - Penalize the violation of the Cauchy Riemann eq. (Least Squares sense)
- Back to the definition of a conformal map
  - For each triangle, define consistently an orthonormal basis (x, y)
  - $\frac{\partial u}{\partial x} = \frac{\partial v}{\partial y} \qquad \frac{\partial u}{\partial y} = -\frac{\partial v}{\partial x}$
  - In other terms $\nabla v = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \nabla u$
  - Let $\nabla u = M(u_0, u_1, u_2)^T$



Back to the roots



- Key idea
  - Penalize the violation of the Cauchy Riemann eq. (Least Squares sense)
- Back to the definition of a conformal map
  - For each triangle, define consistently an orthonormal basis (x, y)
  - $\frac{\partial u}{\partial x} = \frac{\partial v}{\partial y} \qquad \frac{\partial u}{\partial y} = -\frac{\partial v}{\partial x}$
  - In other terms $\nabla v = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \nabla u$
  - Let $\nabla u = M(u_0, u_1, u_2)^T$
- Minimizing the violation of Cauchy Riemann equations
  - $E(\phi) = \sum_t A_t ||M(v_0 v_1 v_2)^T - \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} M(u_0 u_1 u_2)^T||^2$
  - Gradient formulation!

Complex formulation

- Concise Cauchy Riemann equation
  - $X = x + iy, \quad U = u + iv$
  - $\frac{\partial X}{\partial u} - i\frac{\partial X}{\partial v} = 0$
  - Implies: $\frac{\partial U}{\partial x} + i\frac{\partial U}{\partial y} = 0$

Complex formulation

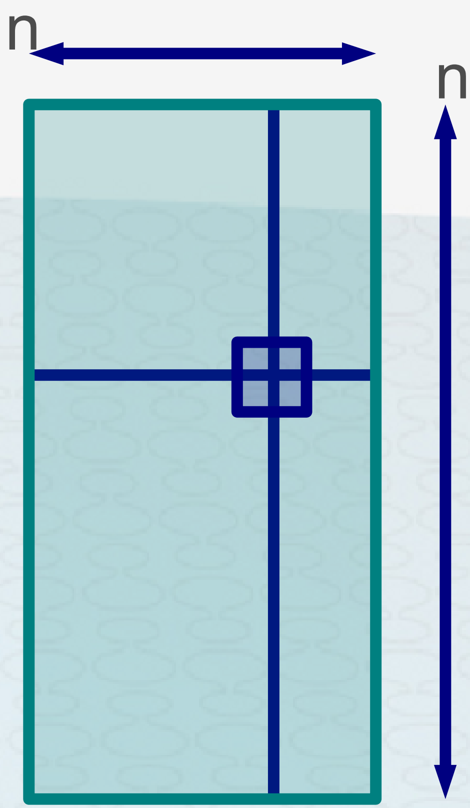
- Concise Cauchy Riemann equation
  - $X = x + iy, \quad U = u + iv$
  - $\frac{\partial X}{\partial u} - i\frac{\partial X}{\partial v} = 0$
  - Implies: $\frac{\partial U}{\partial x} + i\frac{\partial U}{\partial y} = 0$
- Complex gradient
  - $\frac{\partial U}{\partial x} + i\frac{\partial U}{\partial y} = \frac{i}{2A_t}(W_0 W_1 W_2)(U_0 U_1 U_2)^T$
  - $W_j = (x_{(j+2)\%3} - x_{(j+1)\%3}) + i(y_{(j+2)\%3} - y_{(j+1)\%3})$

Complex formulation

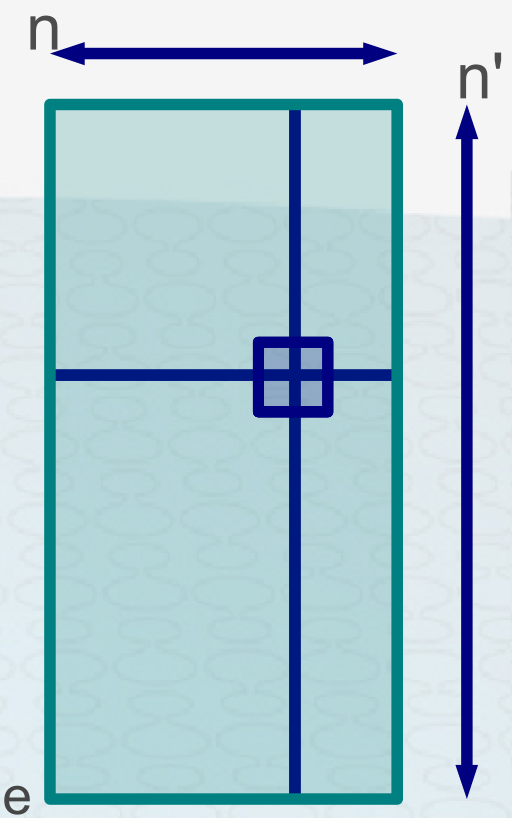
- Concise Cauchy Riemann equation
  - $X = x + iy, \quad U = u + iv$
  - $\frac{\partial X}{\partial u} - i\frac{\partial X}{\partial v} = 0$
  - Implies: $\frac{\partial U}{\partial x} + i\frac{\partial U}{\partial y} = 0$
- Complex gradient
  - $\frac{\partial U}{\partial x} + i\frac{\partial U}{\partial y} = \frac{i}{2A_t}(W_0 W_1 W_2)(U_0 U_1 U_2)^T$
  - $W_j = (x_{(j+2)\%3} - x_{(j+1)\%3}) + i(y_{(j+2)\%3} - y_{(j+1)\%3})$
- Conformal energy
  - $E(U) = \sum_t \frac{1}{2A_t}|(W_0 W_1 W_2)(U_0 U_1 U_2)^T|^2$

Minimizing the conformal energy

- Matrix form
  - $E(U) = U^* C U$
  - U^* : Hermitian complex conjugate
    - $U^*_{i,j} = \overline{U_{j,i}}$

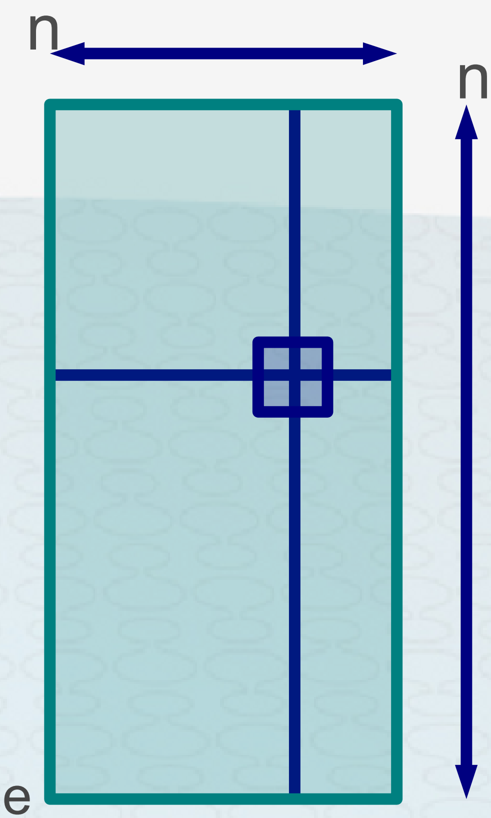


Minimizing the conformal energy



- Matrix form
  - $E(U) = U^* C U$
  - U^* : Hermitian complex conjugate
    - $U^*_{i,j} = \overline{U_{j,i}}$
  - $C = M^* M$
    - M : n'xn matrix (n: vertices, n': triangles)
    - C : nxn matrix
    - $M_{ij} = \frac{(W_{j,t_i})}{\sqrt{2A_{t_i}}}$
    - If the vertex j belongs to triangle i, 0 otherwise

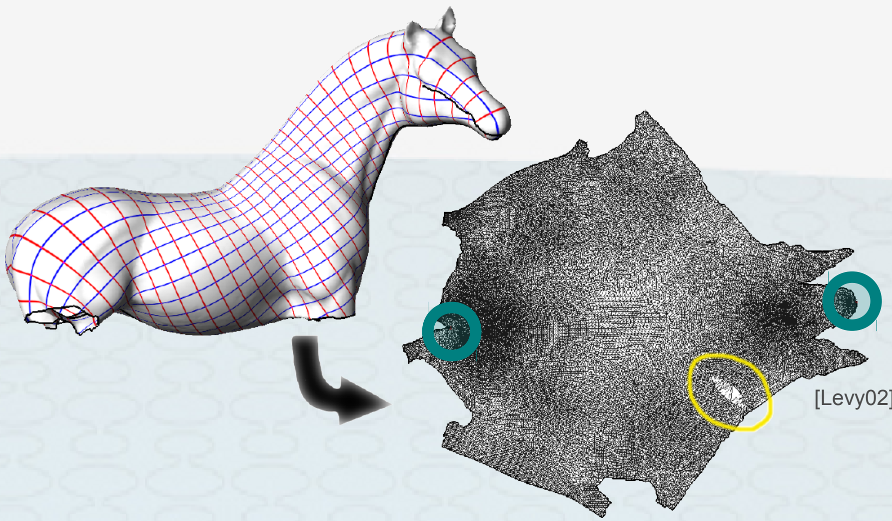
Minimizing the conformal energy



- Matrix form
  - $E(U) = U^*CU$
  - U^* : Hermitian complex conjugate
    - $U^*_{i,j} = \overline{U_{j,i}}$
  - $C = M^*M$
    - M : n'xn matrix (n: vertices, n': triangles)
    - C : nxn matrix
    - $M_{ij} = \frac{(W_{j,t_i})}{\sqrt{2A_{t_i}}}$
    - If the vertex j belongs to triangle i, 0 otherwise
- $E(U) = U^*M^*MU = ||MU||^2$
- Issues with trivial solutions

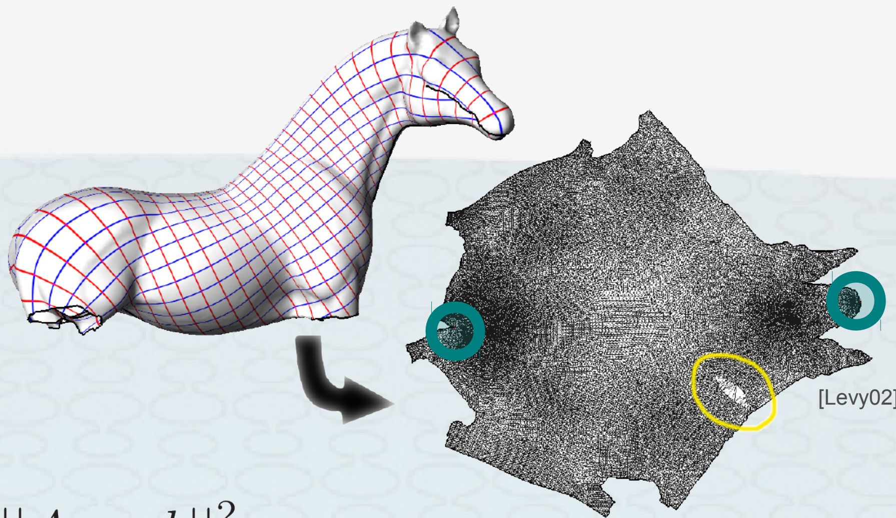
Locking degrees of freedom in a least squares problem

- Avoiding trivial solutions
  - Pinning 2 vertices is enough!
  - Geodesic diameter



[Levy02]

Locking degrees of freedom in a least squares problem

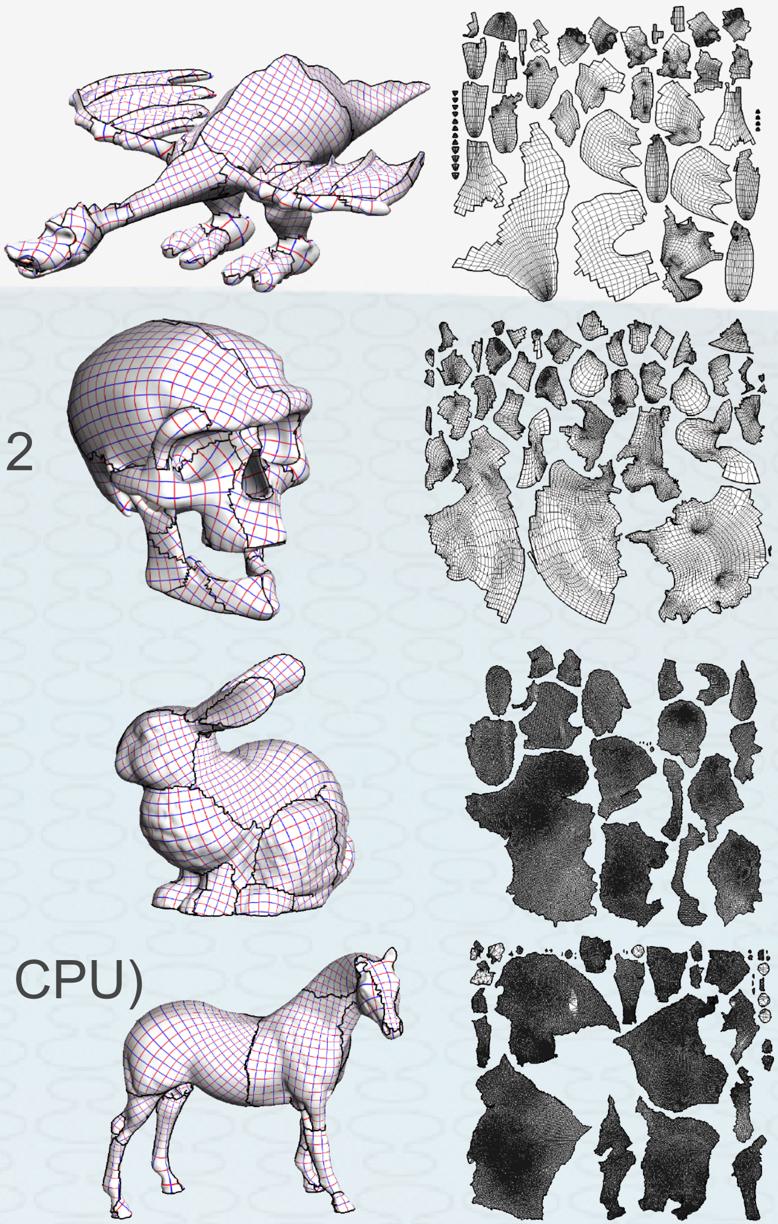


- Avoiding trivial solutions
  - Pinning 2 vertices is enough!
  - Geodesic diameter
- Locking variables
  - $E(U) = ||MU||^2$
  - $E(U) = ||M_F U_F + M_L U_L||^2 = ||Ax - b||^2$
  - M_F : n'x(n – p) matrix, M_L: n'xp, U_F(n-p) vector, U_L p vector
  - $x = (A^T A)^{-1} A^T b$
  - $A = \begin{pmatrix} M_F^{Re} & -M_F^{Im} \\ M_F^{Im} & M_F^{Re} \end{pmatrix}$
  - $b = \begin{pmatrix} M_L^{Re} & -M_L^{Im} \\ M_L^{Im} & M_L^{Re} \end{pmatrix} \begin{pmatrix} U_L^{Re} \\ U_L^{Im} \end{pmatrix}$

A few results

- Theoretical results
  - The matrix A has full rank with p $\geq$ 2
  - The solution is indeed unique
  - Solution independent of the quality of the input triangulation

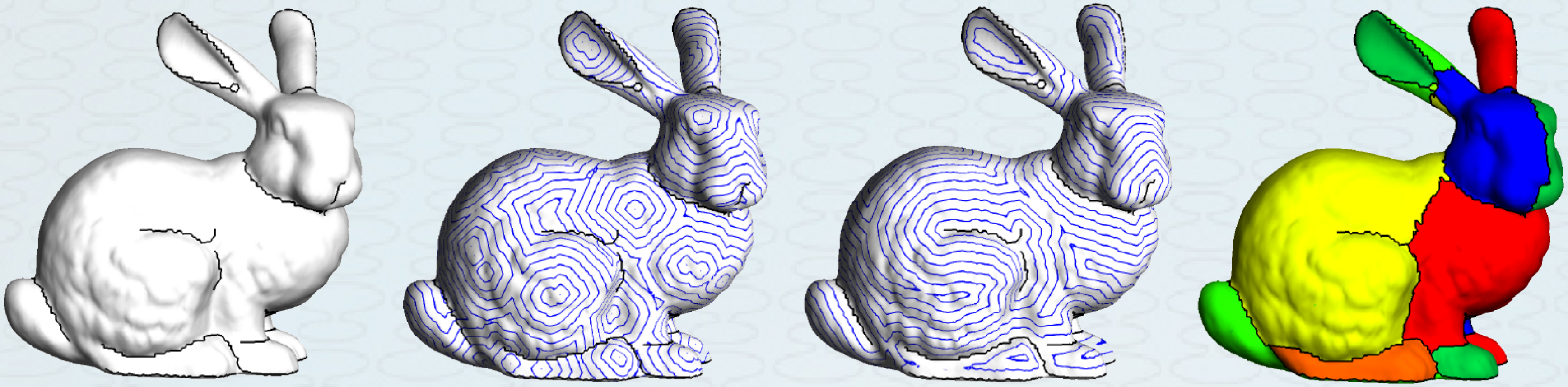
- Practical results
  - Solver: conjugate gradient
  - At the time, dozens of seconds (P3 CPU)
  - Very low angular distortion



[Levy02]

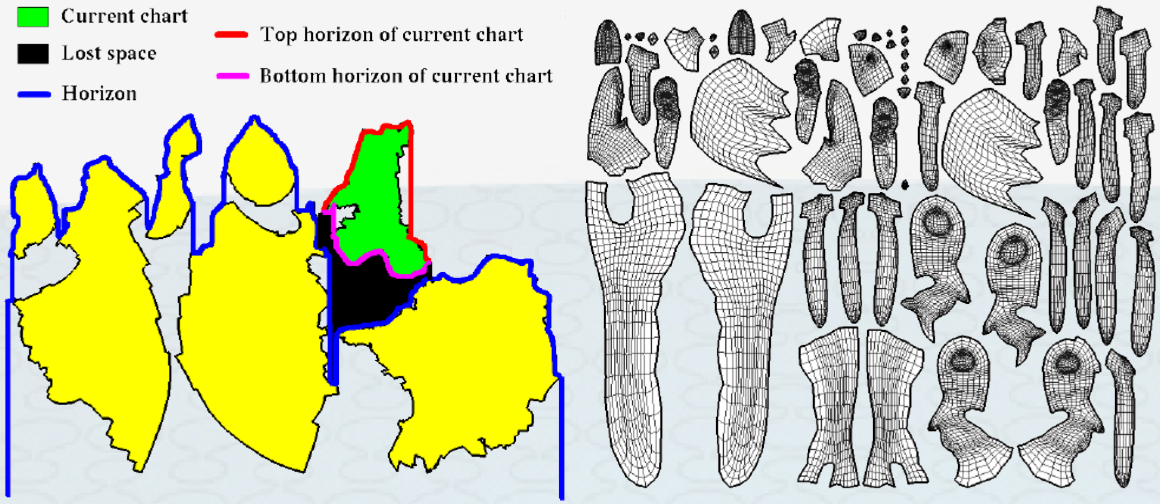
Automatic atlas generation

- Hold on...
  - The input surface has to be homeomorphic to a disc...
  - Let's partition it into disc segments (usually done manually)
  - Hide discontinuities in concave configurations (normals)
  - Geodesic distance from the feature lines: seed extraction
  - Chart merging if the contact point is too early
  - No guarantee on the induced distortion, see [Wang 2008]



[Levy02]

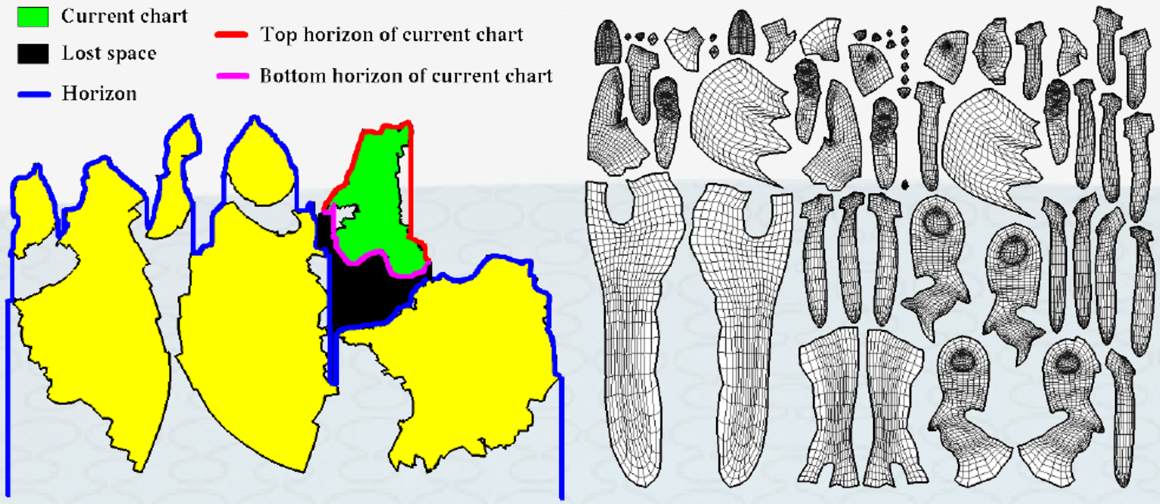
Texture packing



[Levy02]

- Making a good usage of the texture memory
  - Maximize the filling of the texture space with non convex polygons
  - Known as the packing problem (NP-complete)

Texture packing

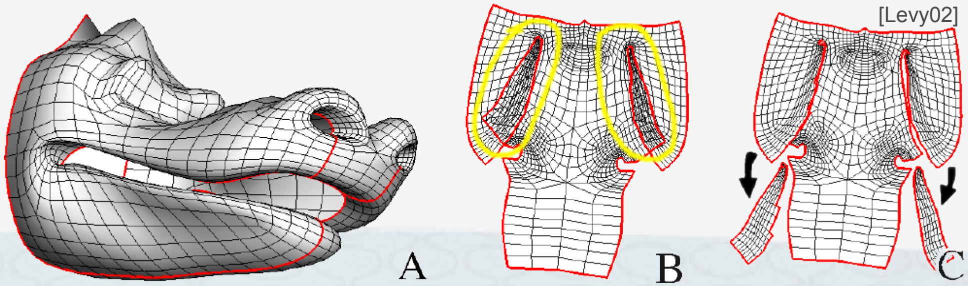


[Levy02]

- Making a good usage of the texture memory
  - Maximize the filling of the texture space with non convex polygons
  - Known as the packing problem (NP-complete)
- A Tetris game
  - Rescale each unfolded chart to its original 3D area
  - Maximum diameter oriented vertically + sorting in decreasing order
  - Horizon computation
  - Minimize the lost space for each chart

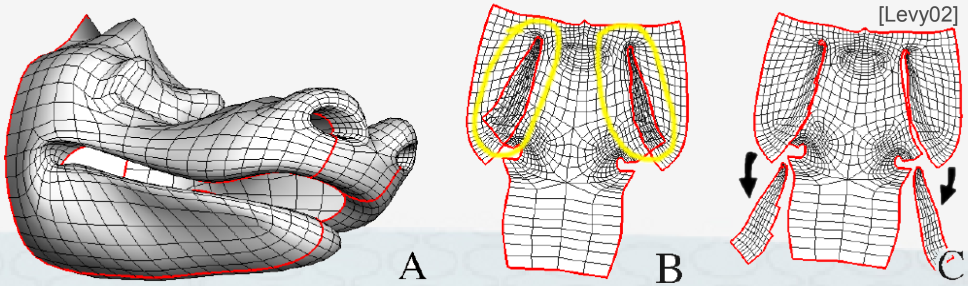
Can we do better?

Can we do better?

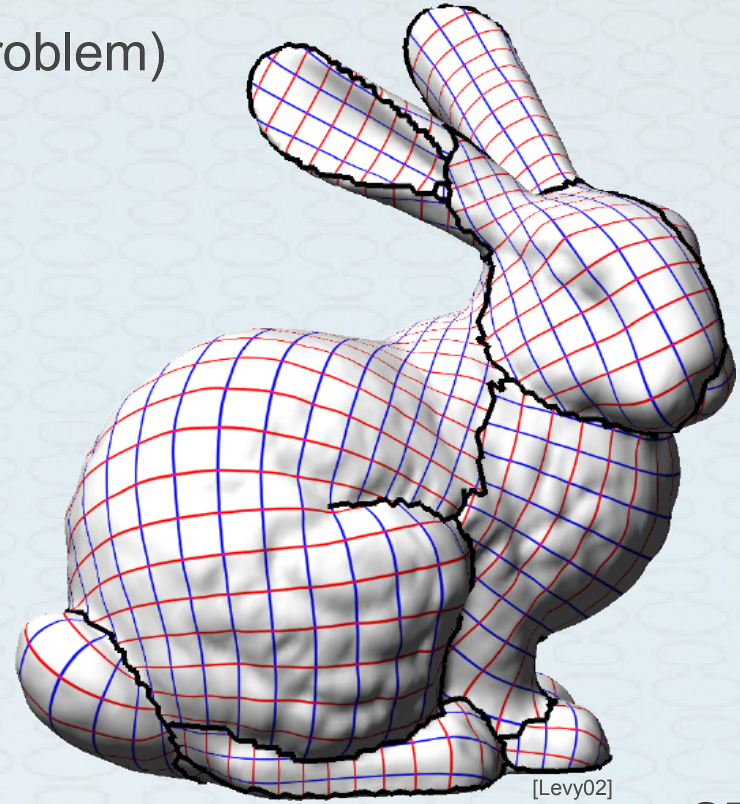


- Technical limitations (~minor)
  - Triangle flips can occur
  - Overlapping can occur (not really a problem)

Can we do better?

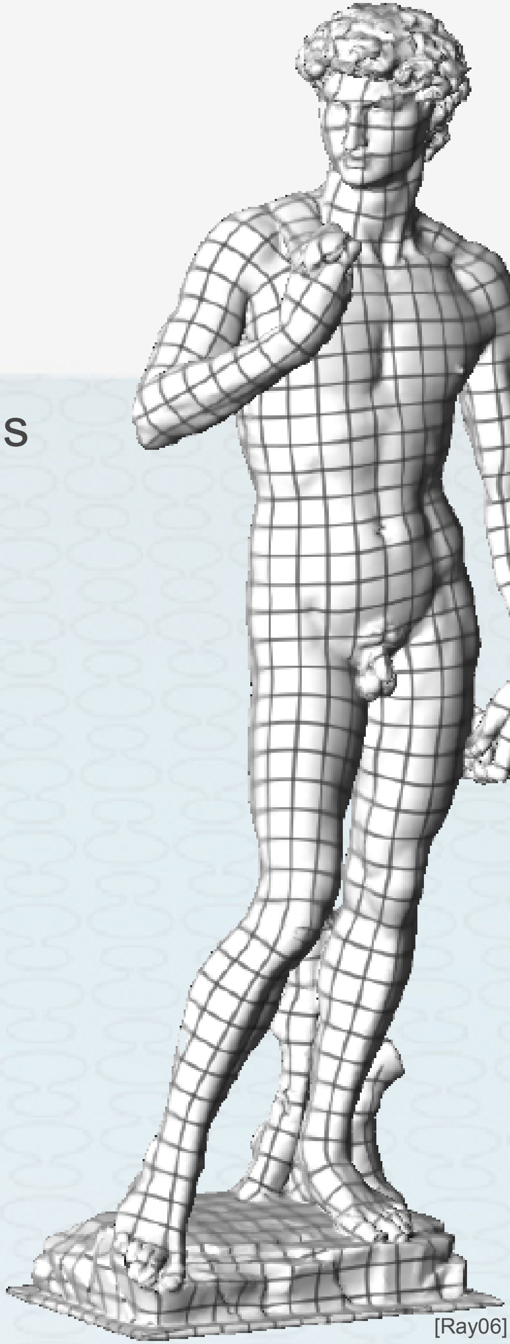


- Technical limitations (~minor)
  – Triangle flips can occur
  – Overlapping can occur (not really a problem)
- More fundamental limitation
  – Important discontinuity across chart boundaries
- Problematic for applications
  – Texturing, just alright: visual artifacts are often hidden by shading
  – What about other signals? (bumps)
- Towards **global** parameterization



Global Parameterization

- Computing unfoldings with global continuity
  - Obviously, only discs unfold to the plane
  - Notion of chart atlas and transition functions
  - Well behaved transition functions



[Ray06]

Global Parameterization

- Computing unfoldings with global continuity
  - Obviously, only discs unfold to the plane
  - Notion of chart atlas and transition functions
  - Well behaved transition functions

- Notion of surface quadrangulation
  - Contouring of global parameterizations
  - Quadrilaterals in place of triangles
  - *Reverse-engineer* the geometric structure
  - More on this later

[Ray06]

Global Parameterization

- Computing unfoldings with global continuity
  - Obviously, only discs unfold to the plane
  - Notion of chart atlas and transition functions
  - Well behaved transition functions
- Notion of surface quadrangulation
  - Contouring of global parameterizations
  - Quadrilaterals in place of triangles
  - *Reverse-engineer* the geometric structure
  - More on this later
- Many (sophisticated) attempts
  - Integrating direction fields
  - [Ray et al. 2006] [Kalberer et al. 2007]
  - Prone to numerical instabilities
  - Why are those solutions sophisticated?

[Ray06]

From local to global parameterization

"Any problem which is non-linear in character, which involves more than one coordinate system (…) is likely to require considerations of topology and group theory for its solution.

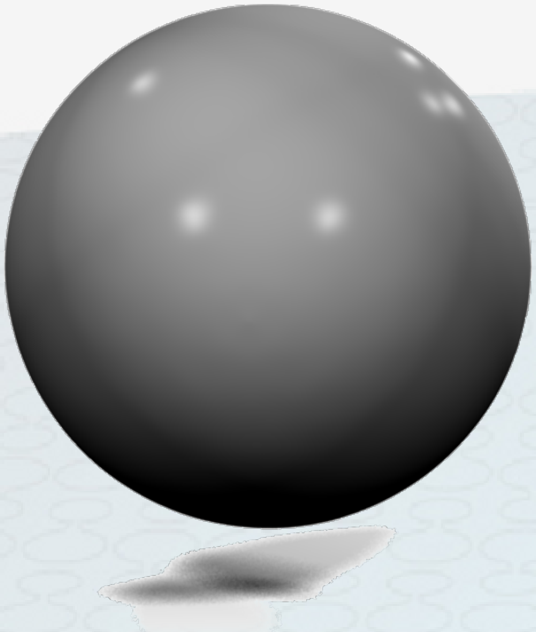
In the solution of such problems, classical analysis will frequently appear as an instrument in the small, integrated over the whole problem with the aid of topology or group theory."

Marston Morse, 1934

Gauss Bonnet theorem revisited (1/2)

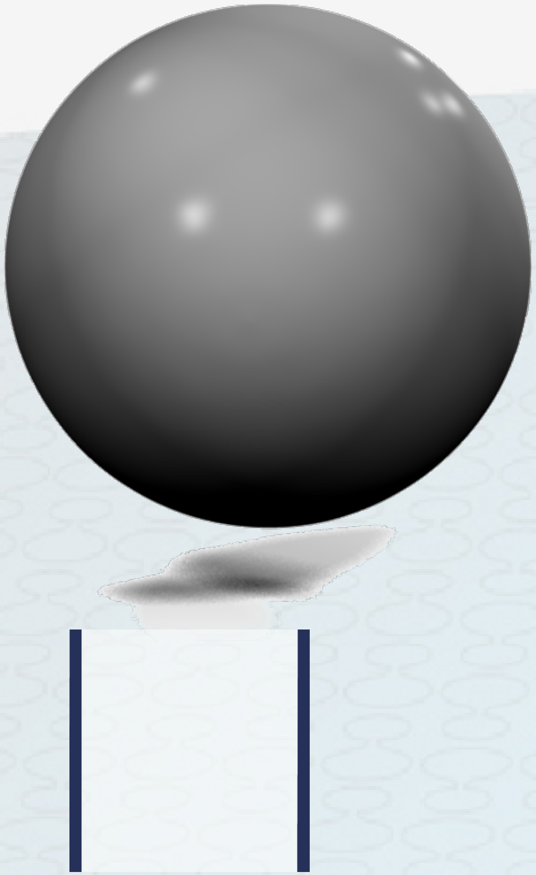
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



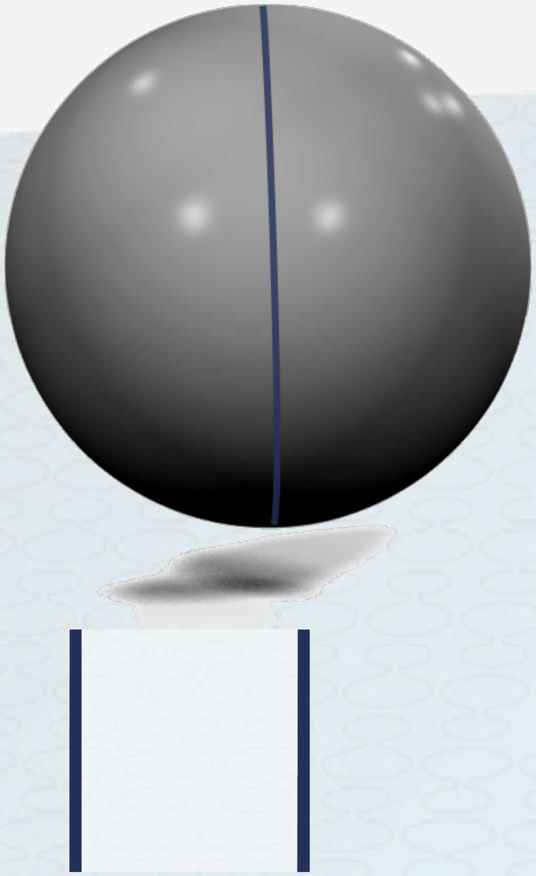
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



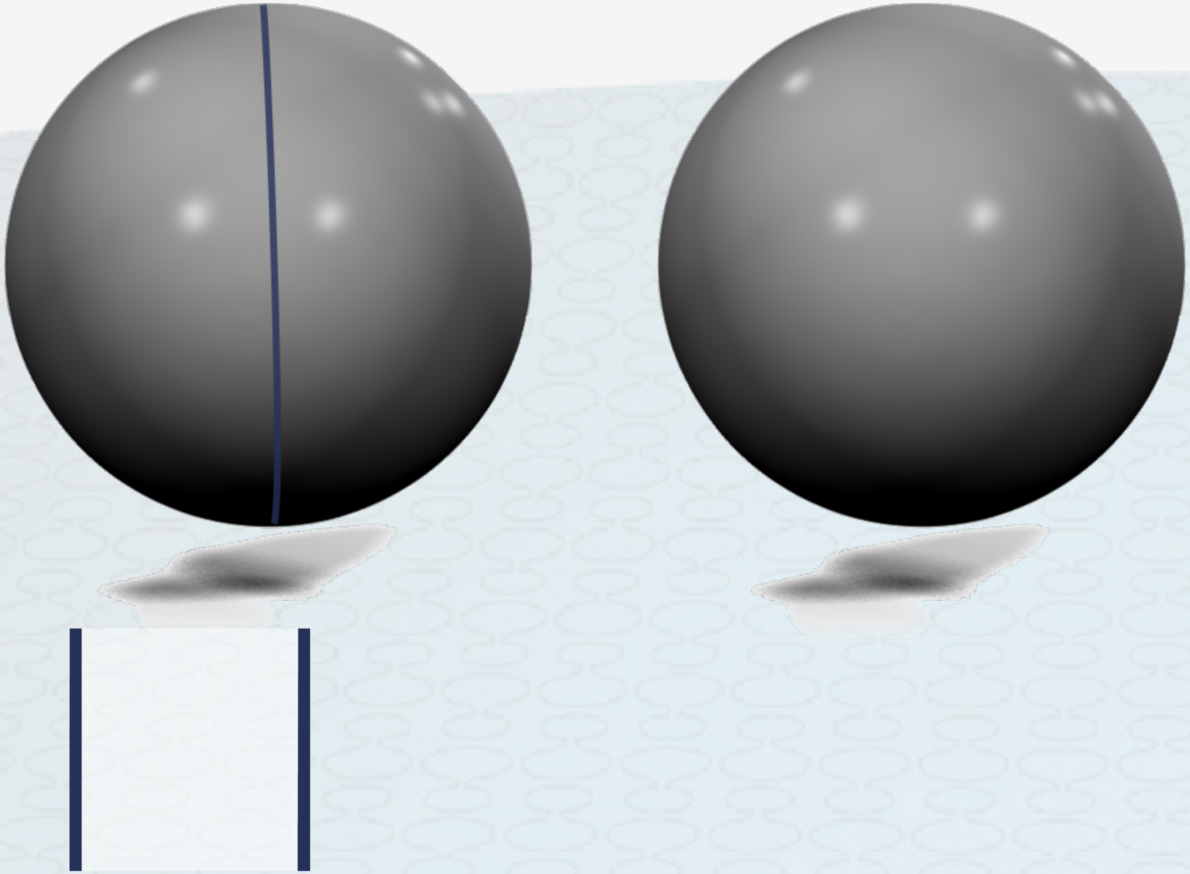
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



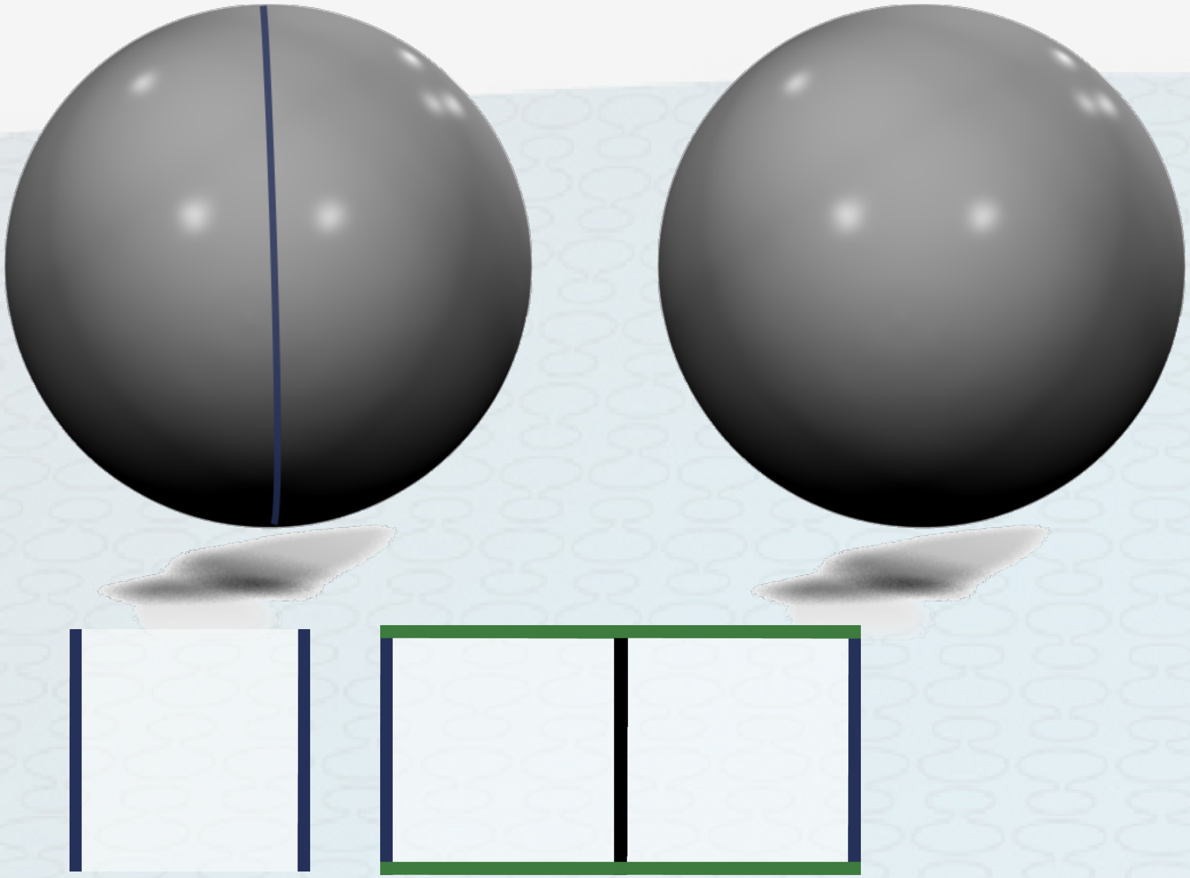
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



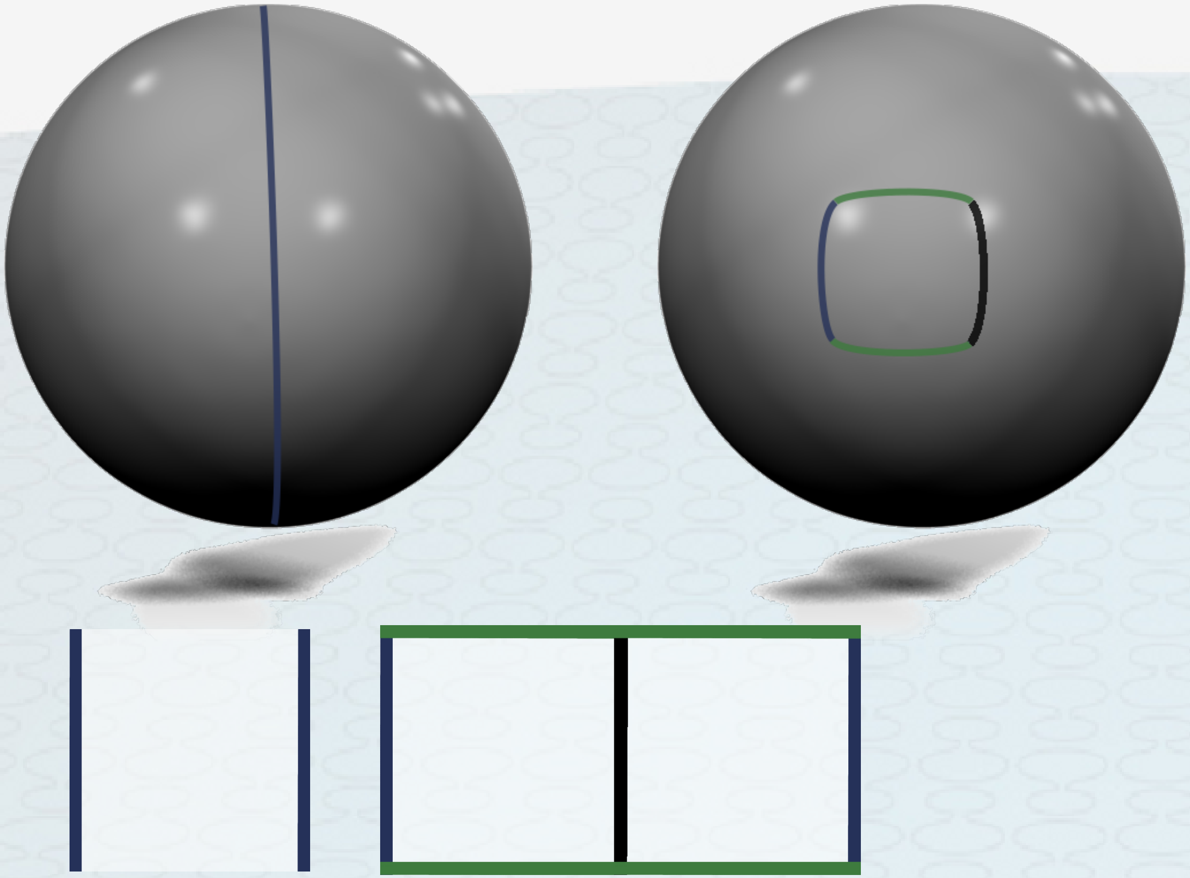
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



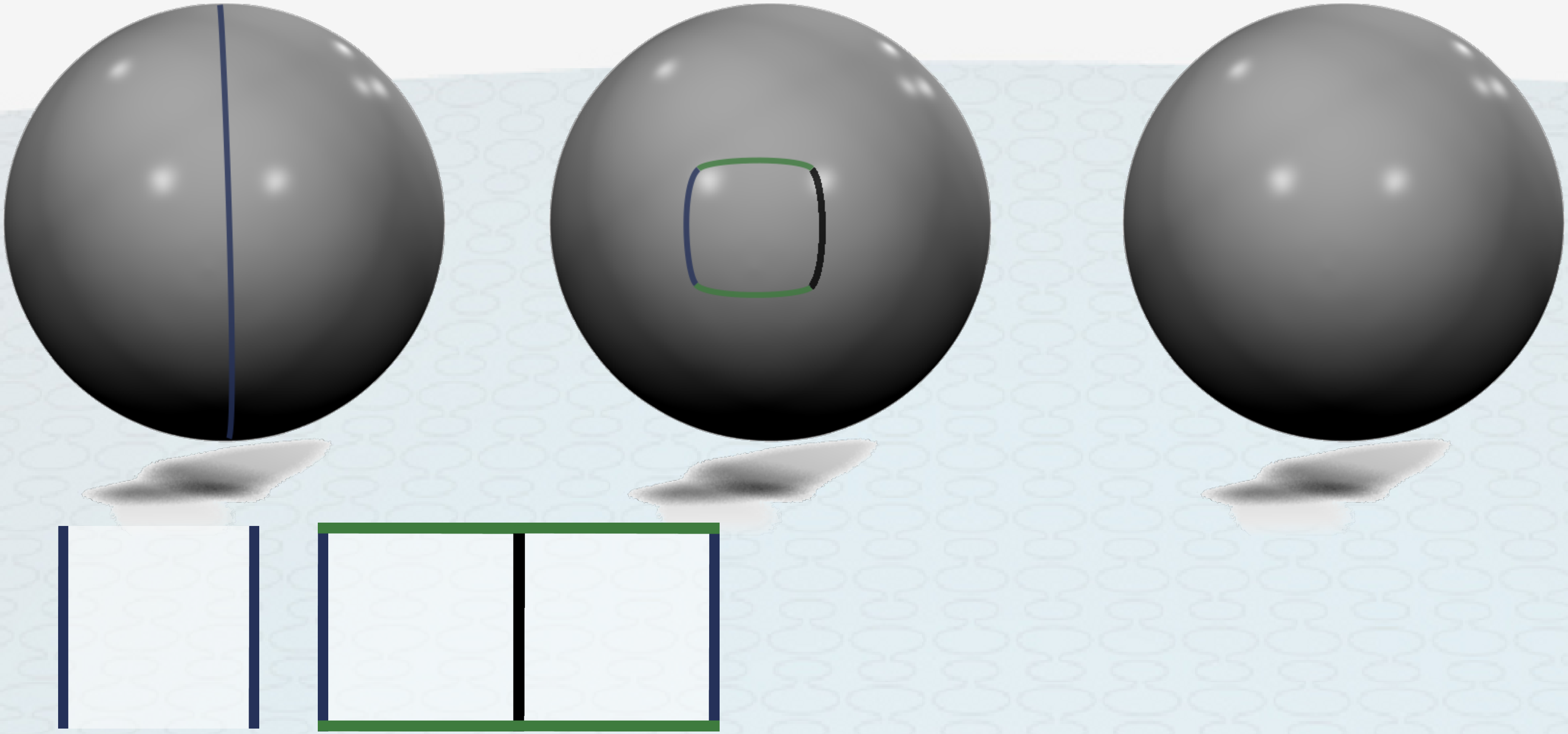
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



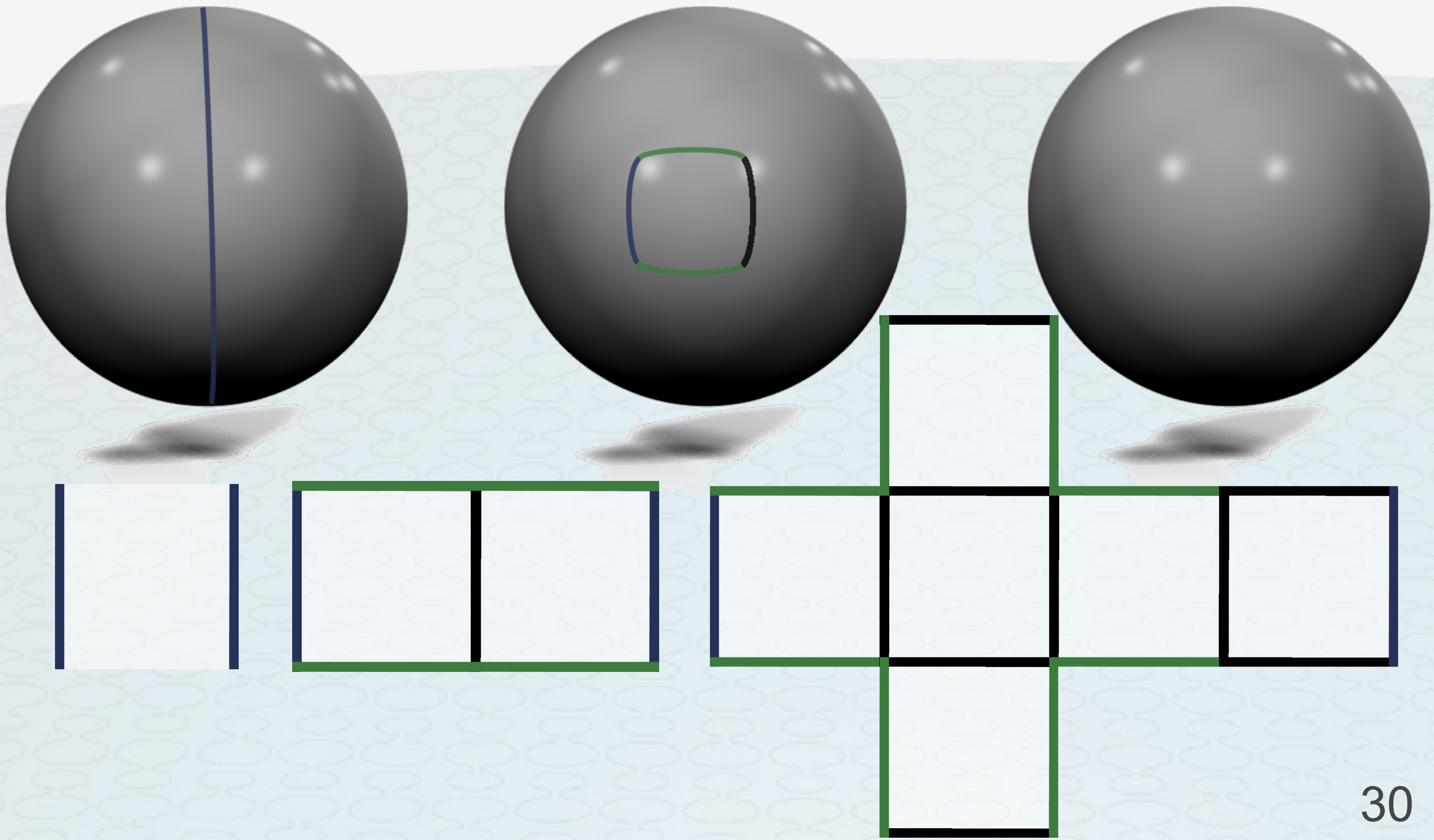
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



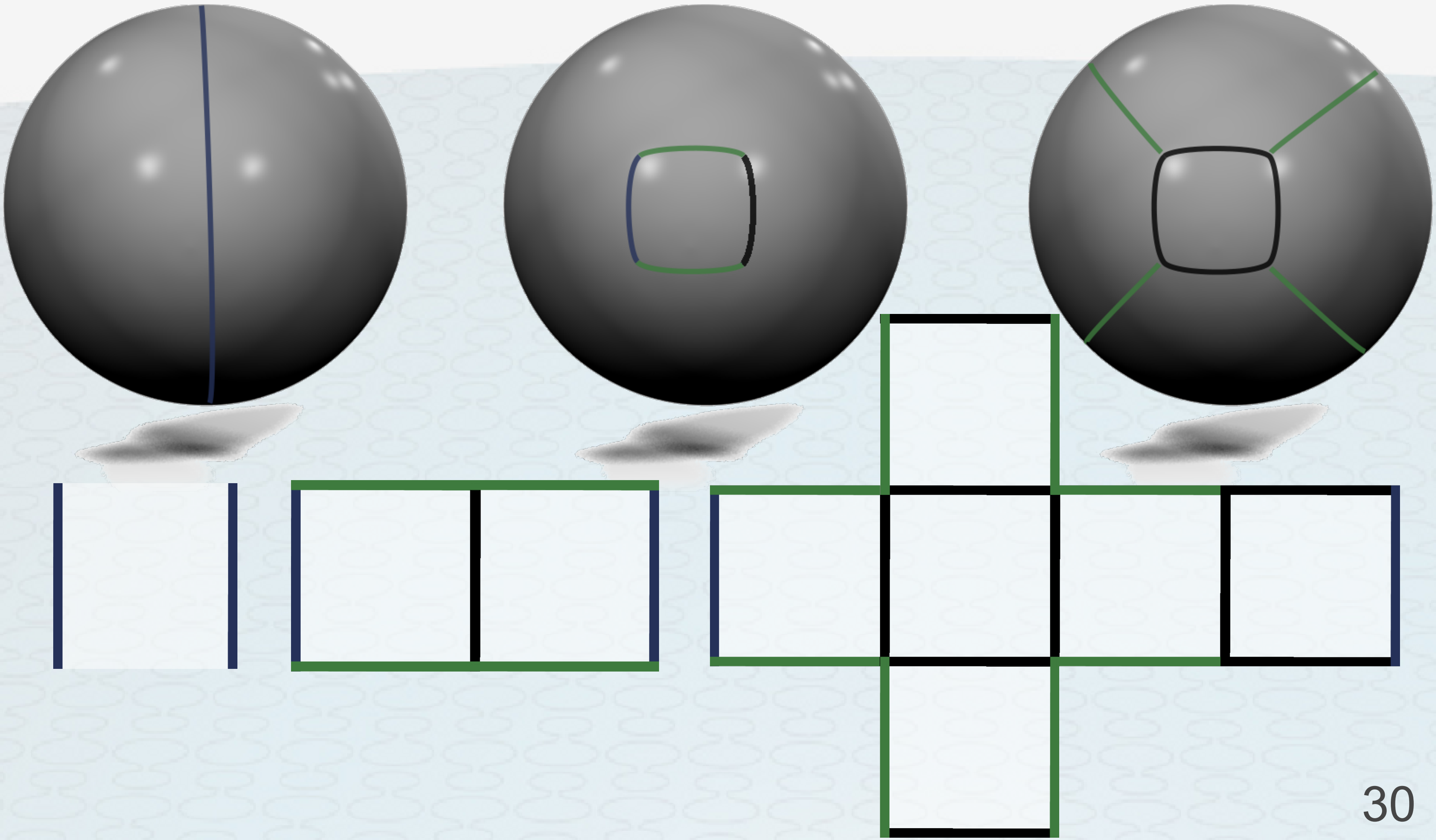
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



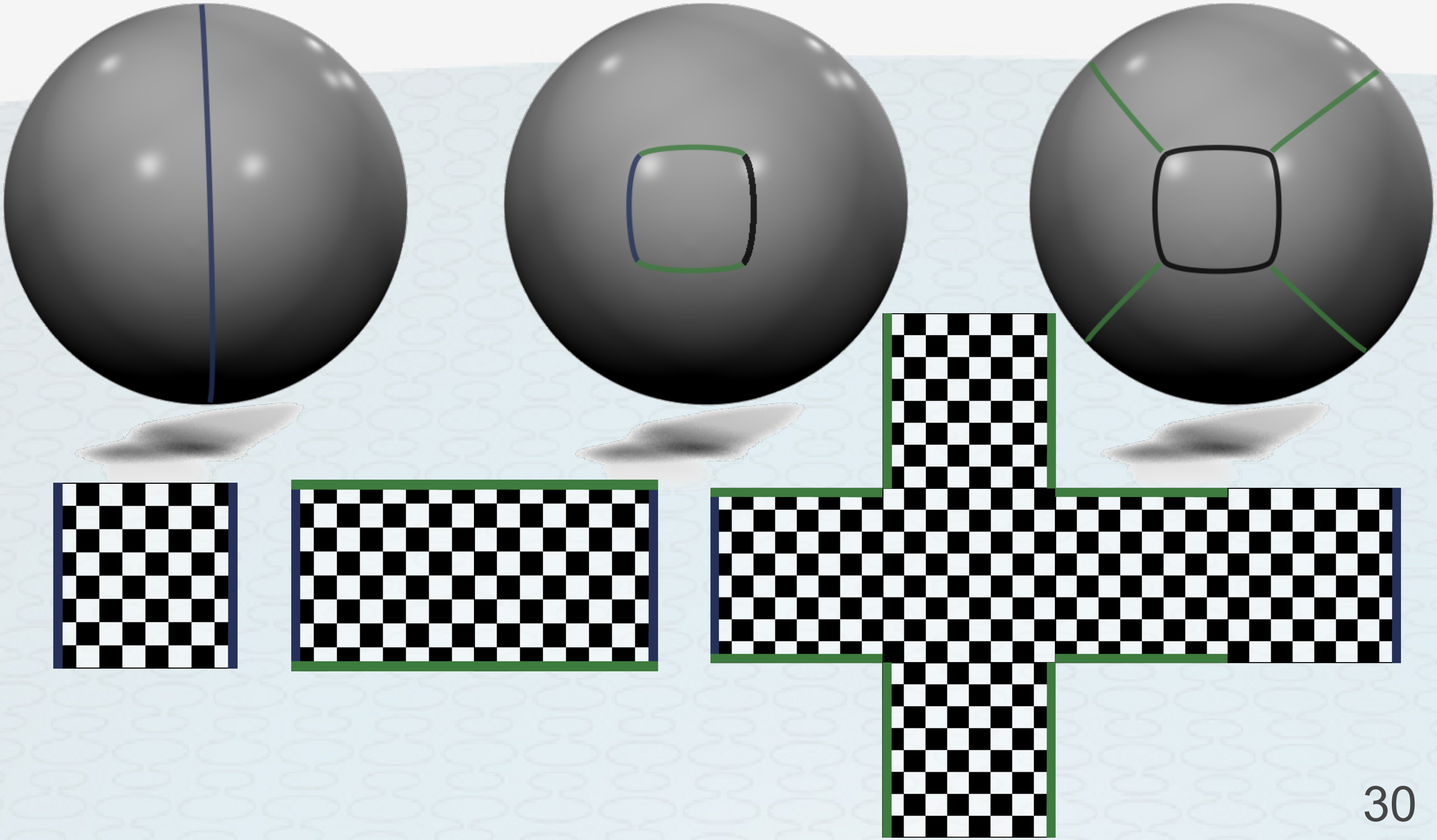
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



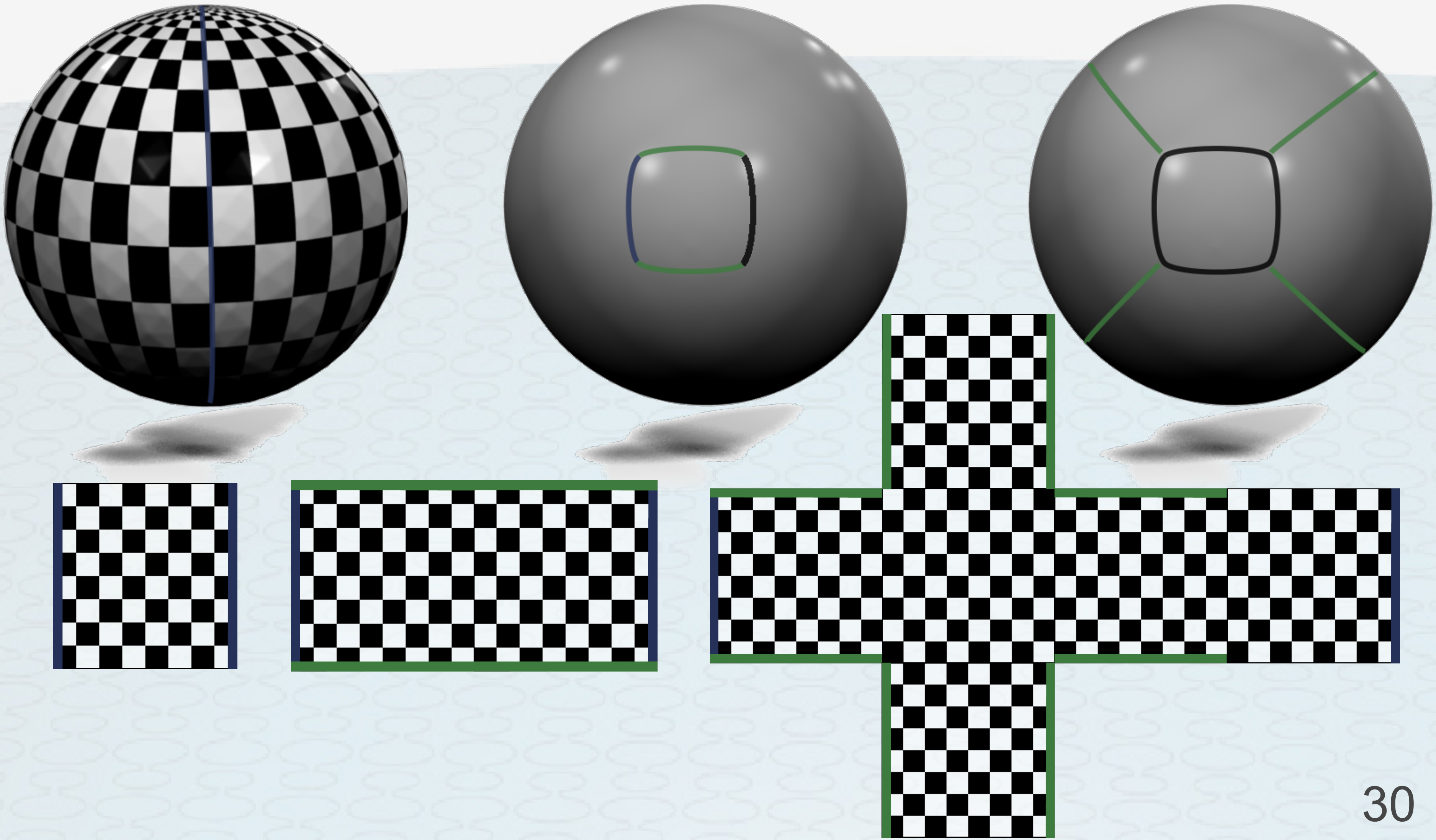
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



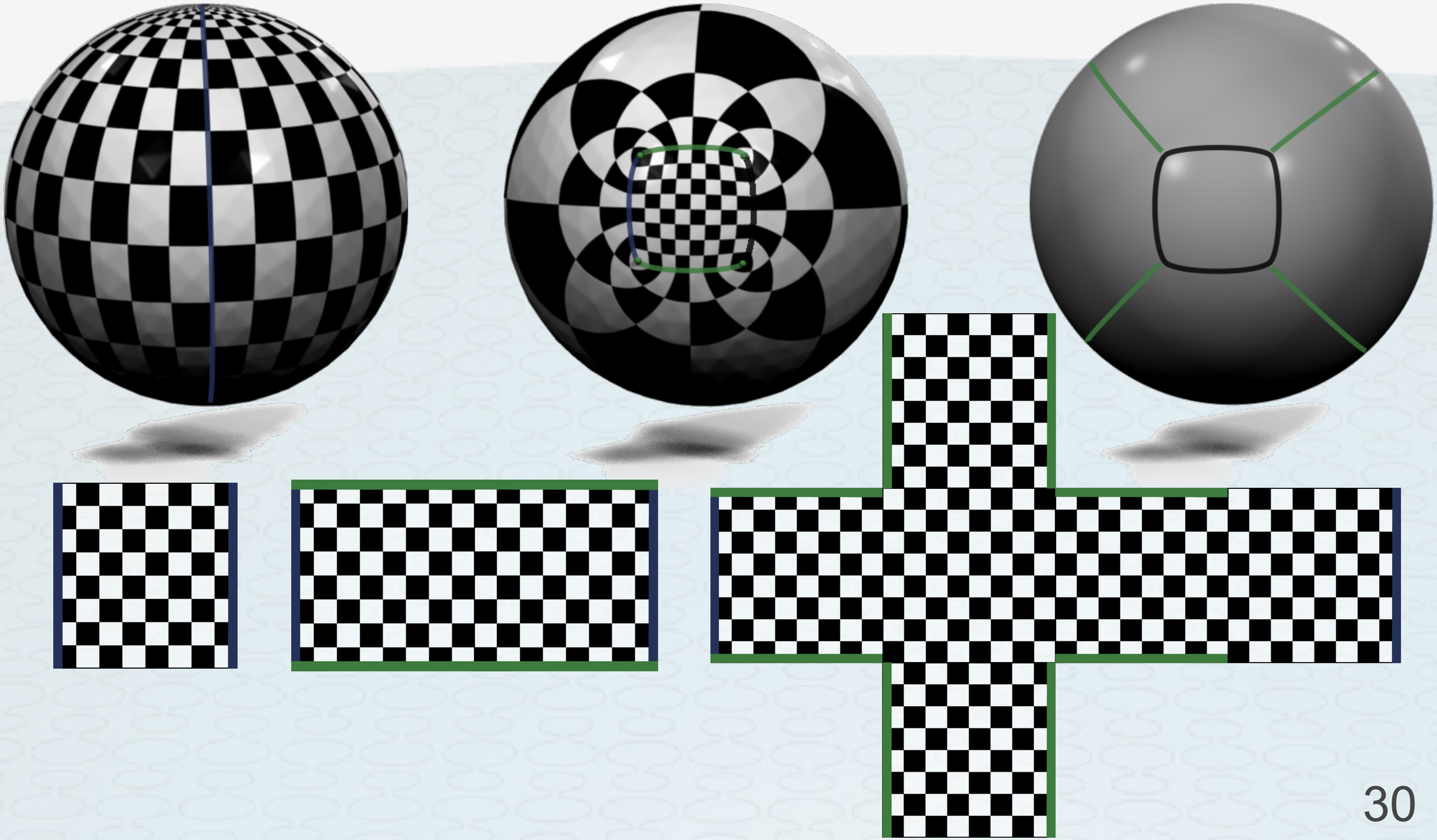
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



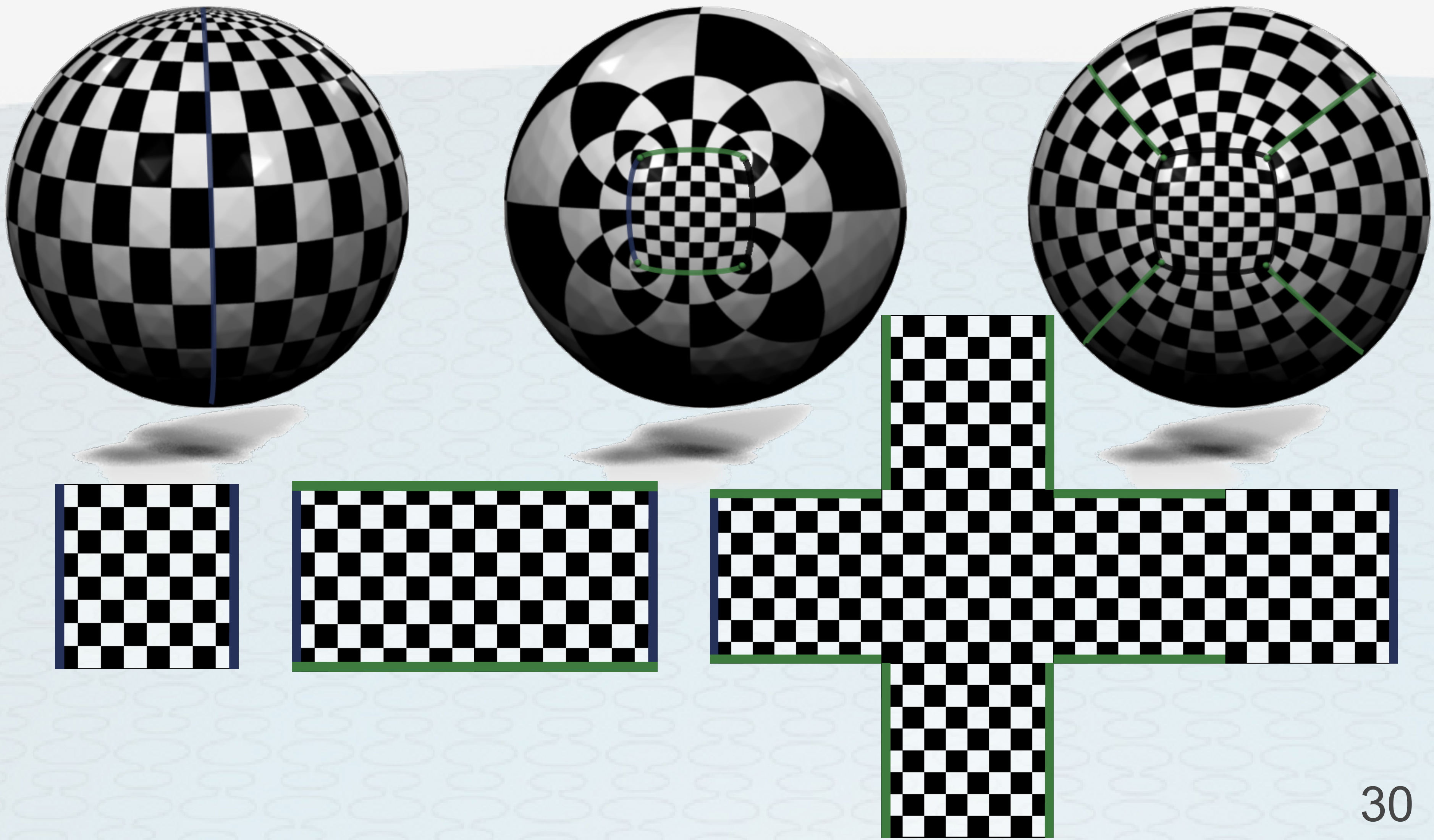
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



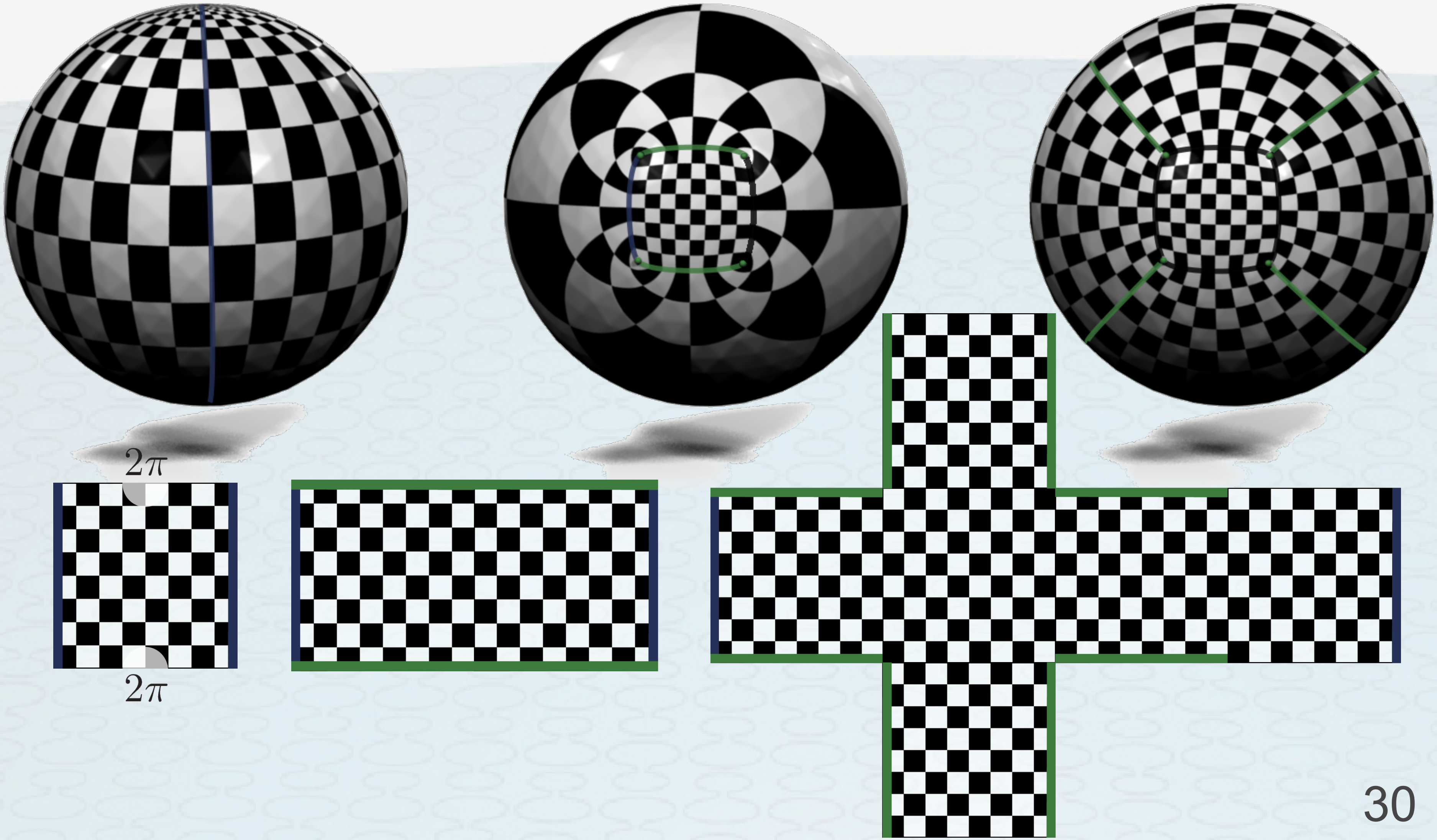
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



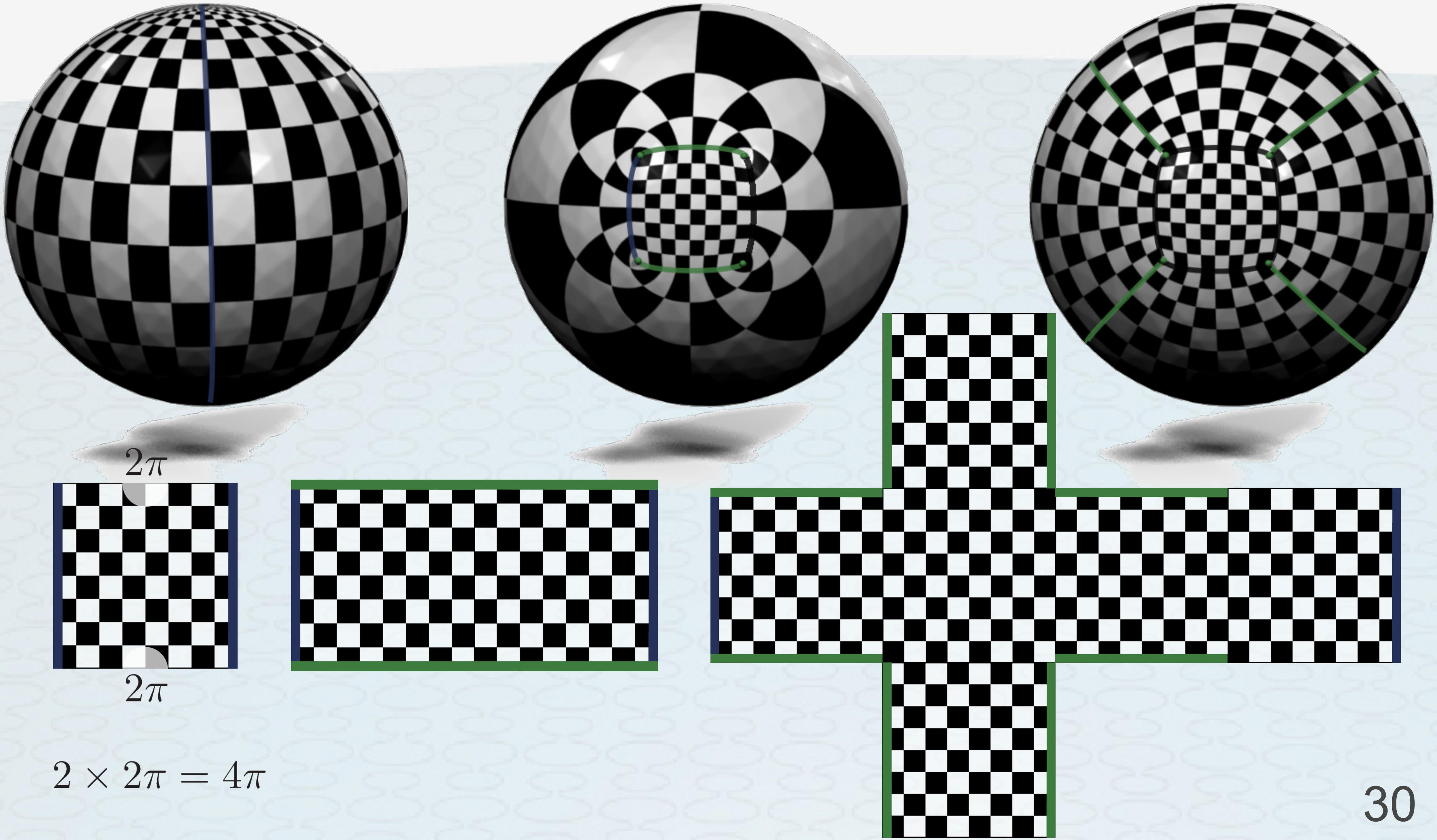
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



Gauss Bonnet theorem revisited (1/2)

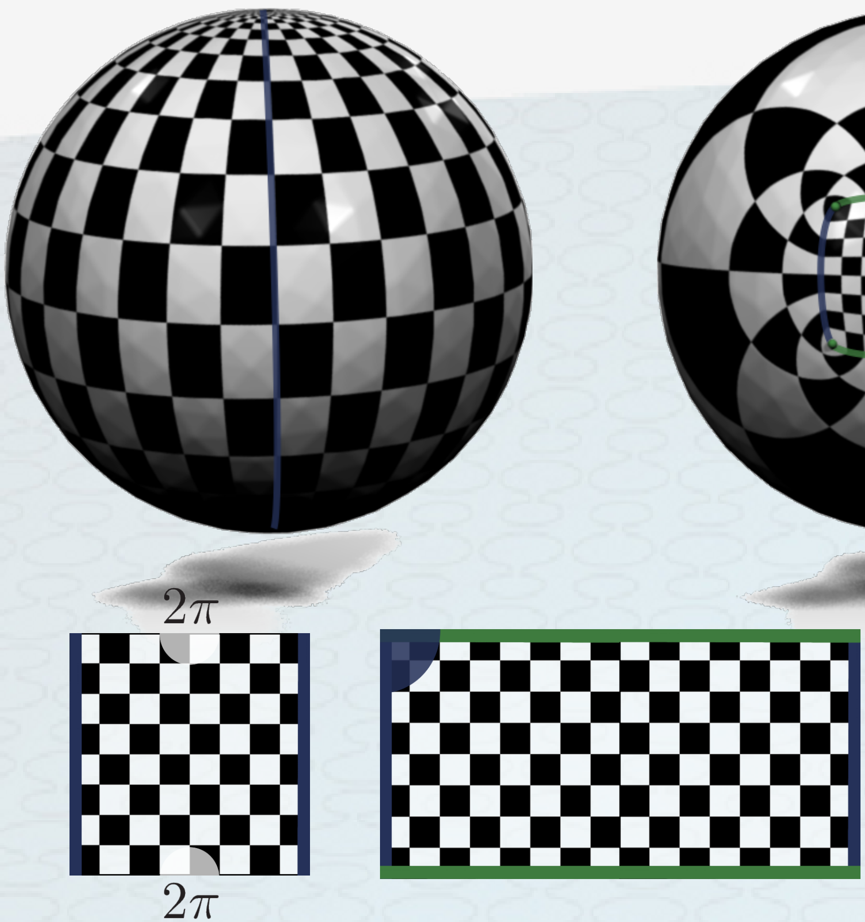
- How many *elastic* quadrilaterals do I need to wrap a sphere?

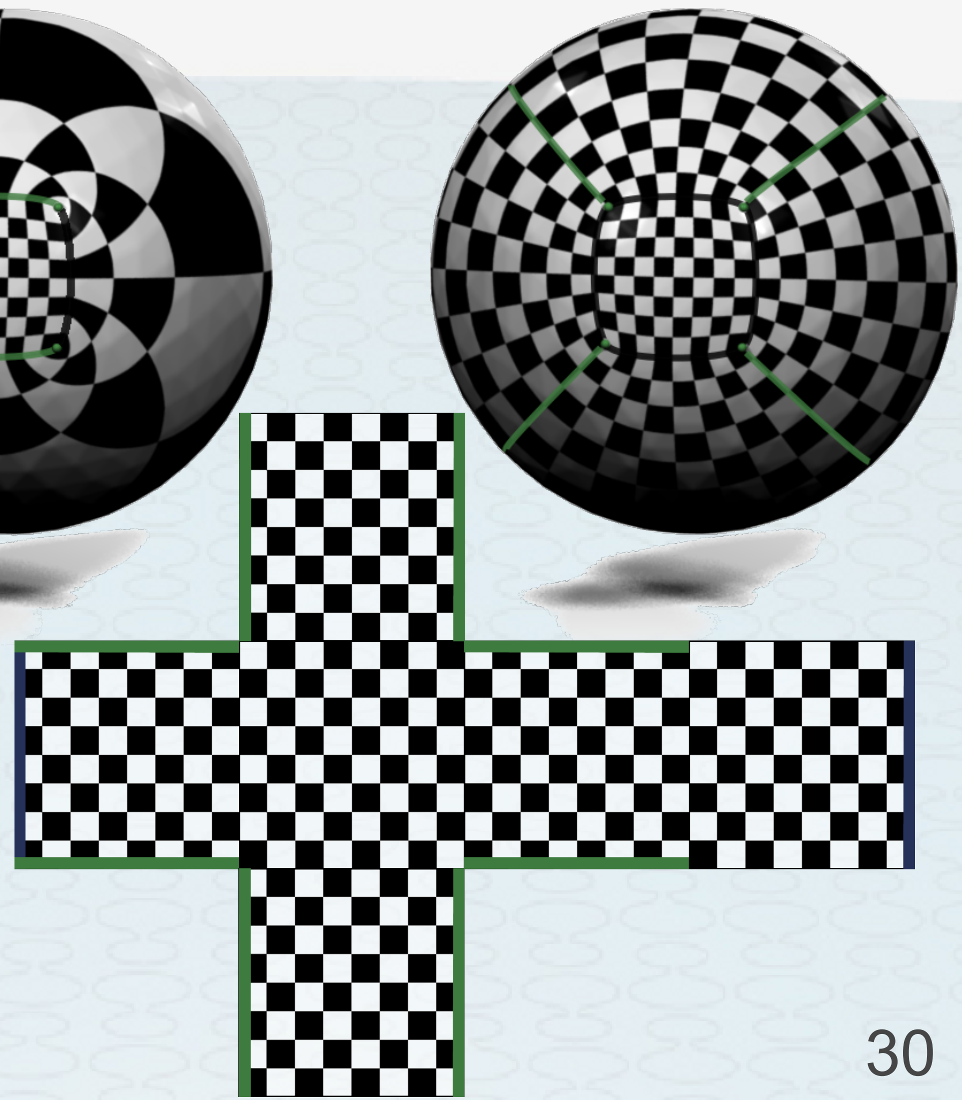


$2 \times 2\pi = 4\pi$

Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?

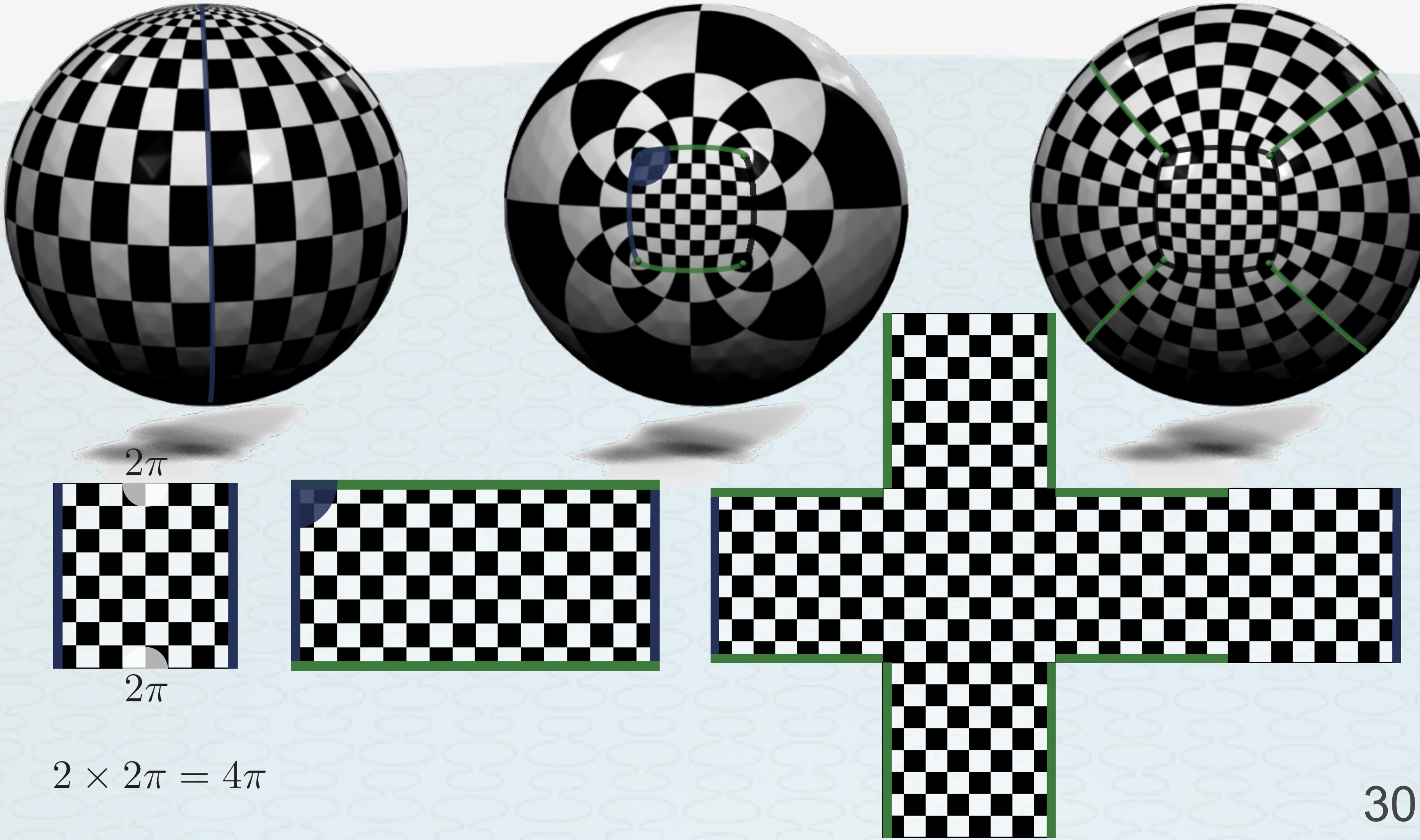




$2 \times 2\pi = 4\pi$

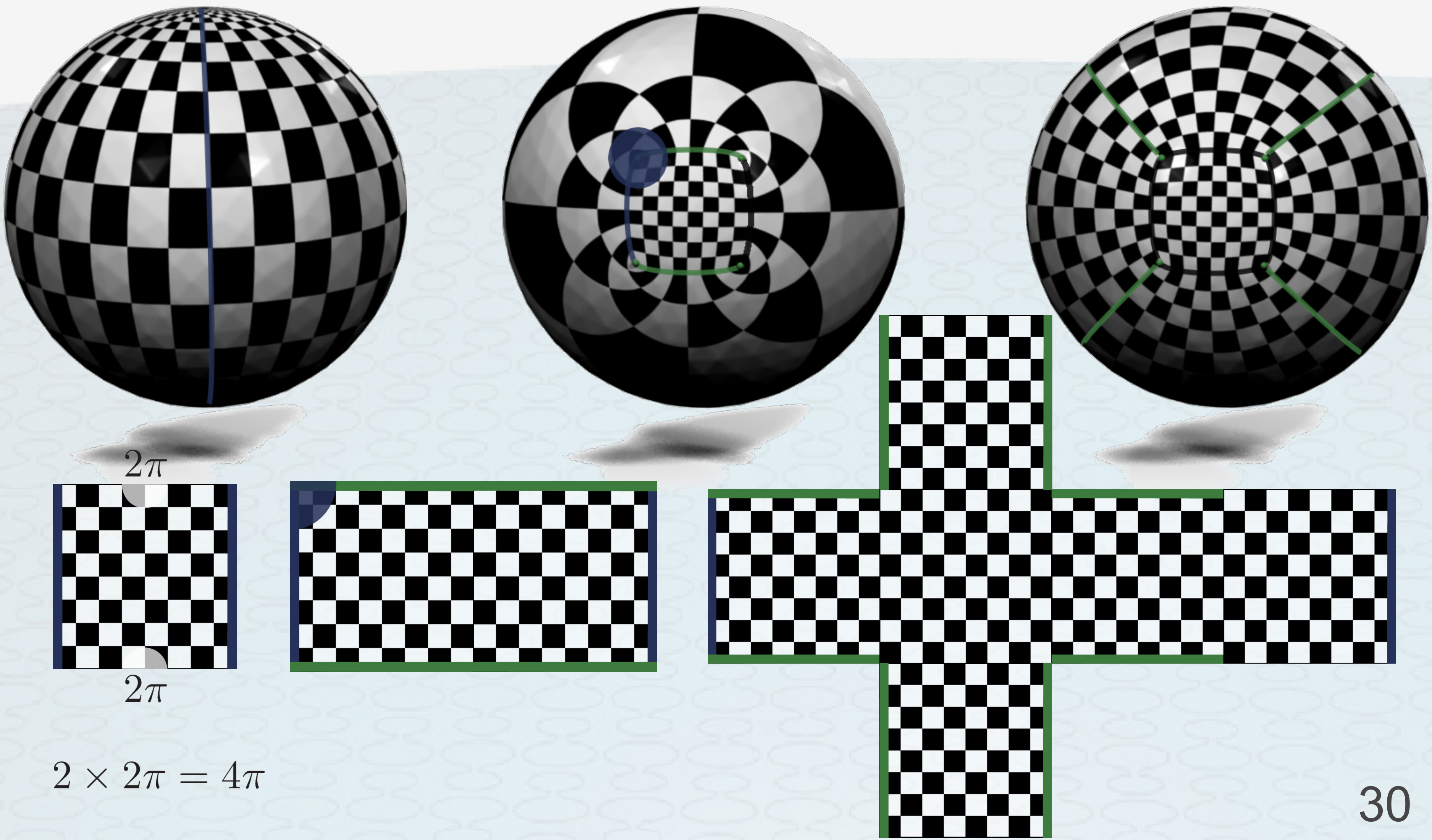
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



Gauss Bonnet theorem revisited (1/2)

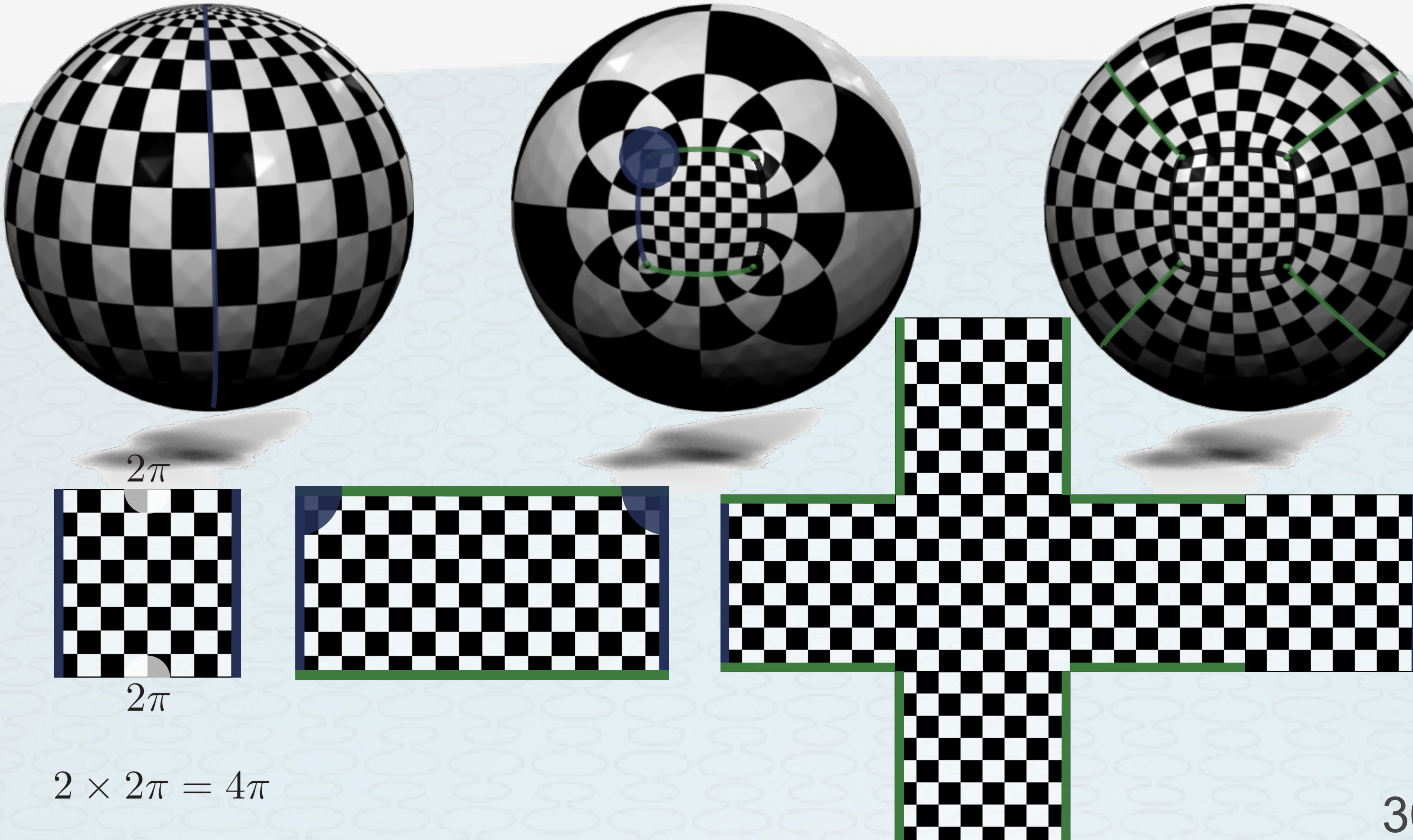
- How many *elastic* quadrilaterals do I need to wrap a sphere?



$$2 \times 2\pi = 4\pi$$

Gauss Bonnet theorem revisited (1/2)

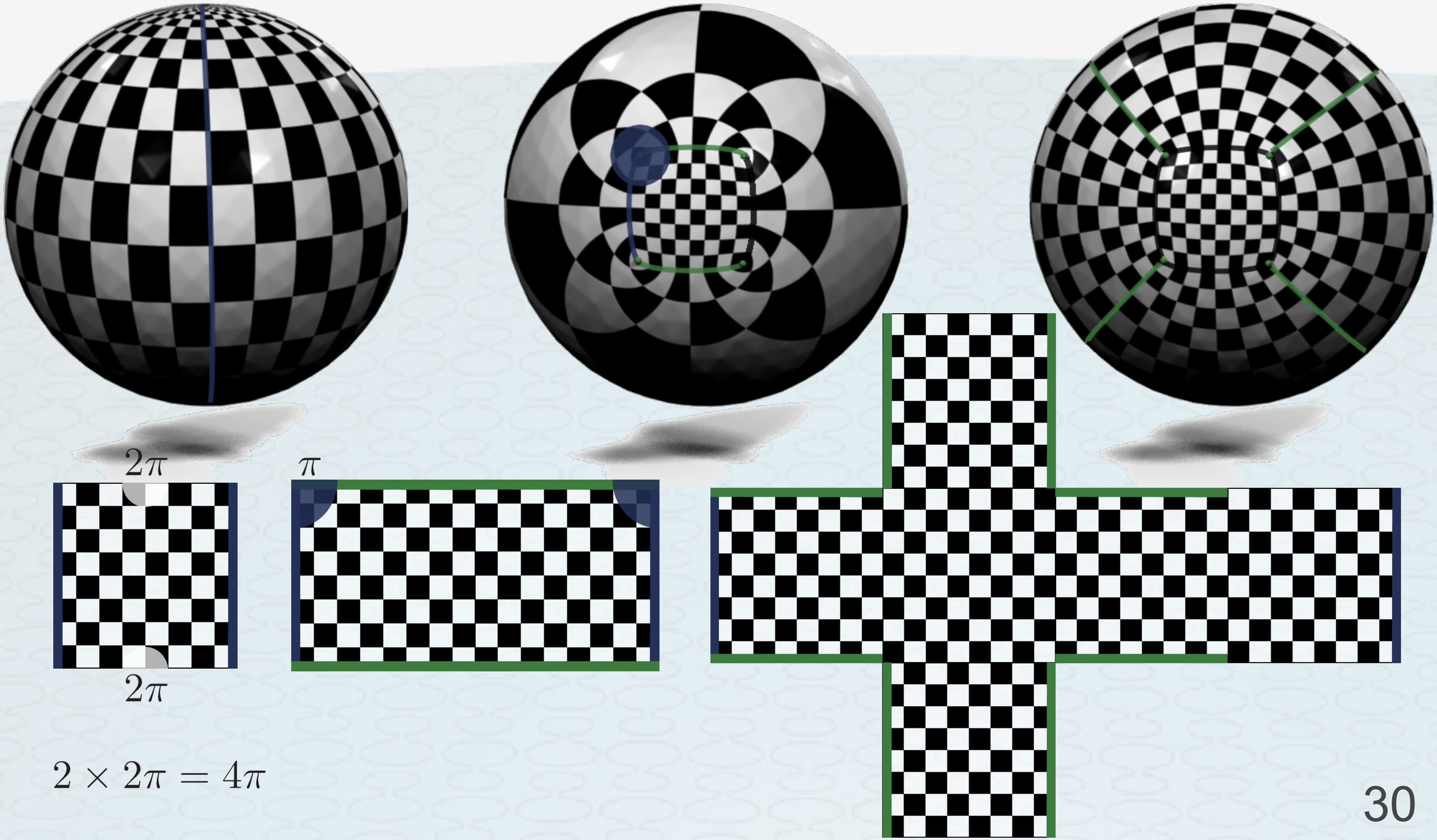
- How many *elastic* quadrilaterals do I need to wrap a sphere?



$$2 \times 2\pi = 4\pi$$

Gauss Bonnet theorem revisited (1/2)

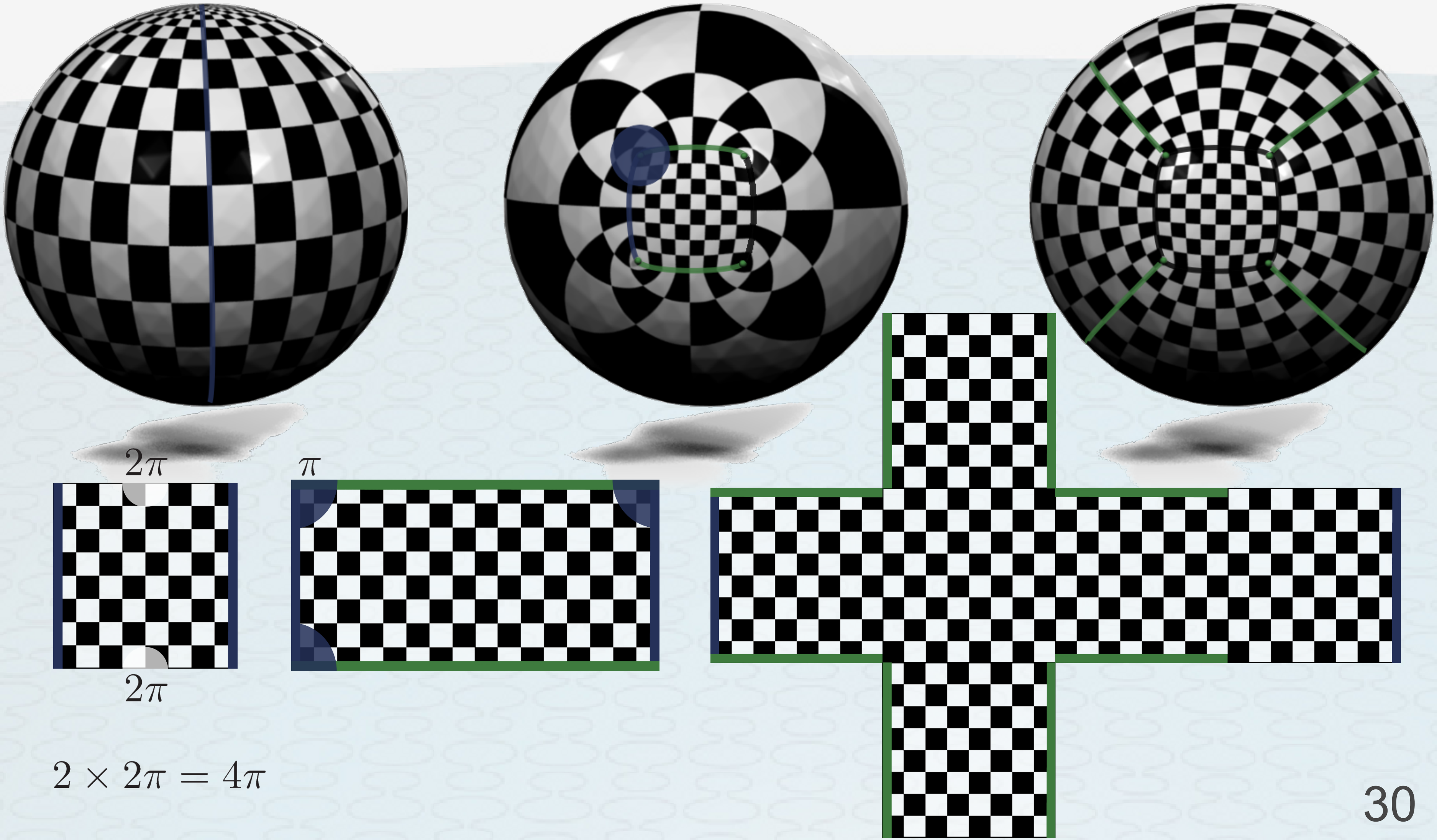
- How many *elastic* quadrilaterals do I need to wrap a sphere?



$2 \times 2\pi = 4\pi$

Gauss Bonnet theorem revisited (1/2)

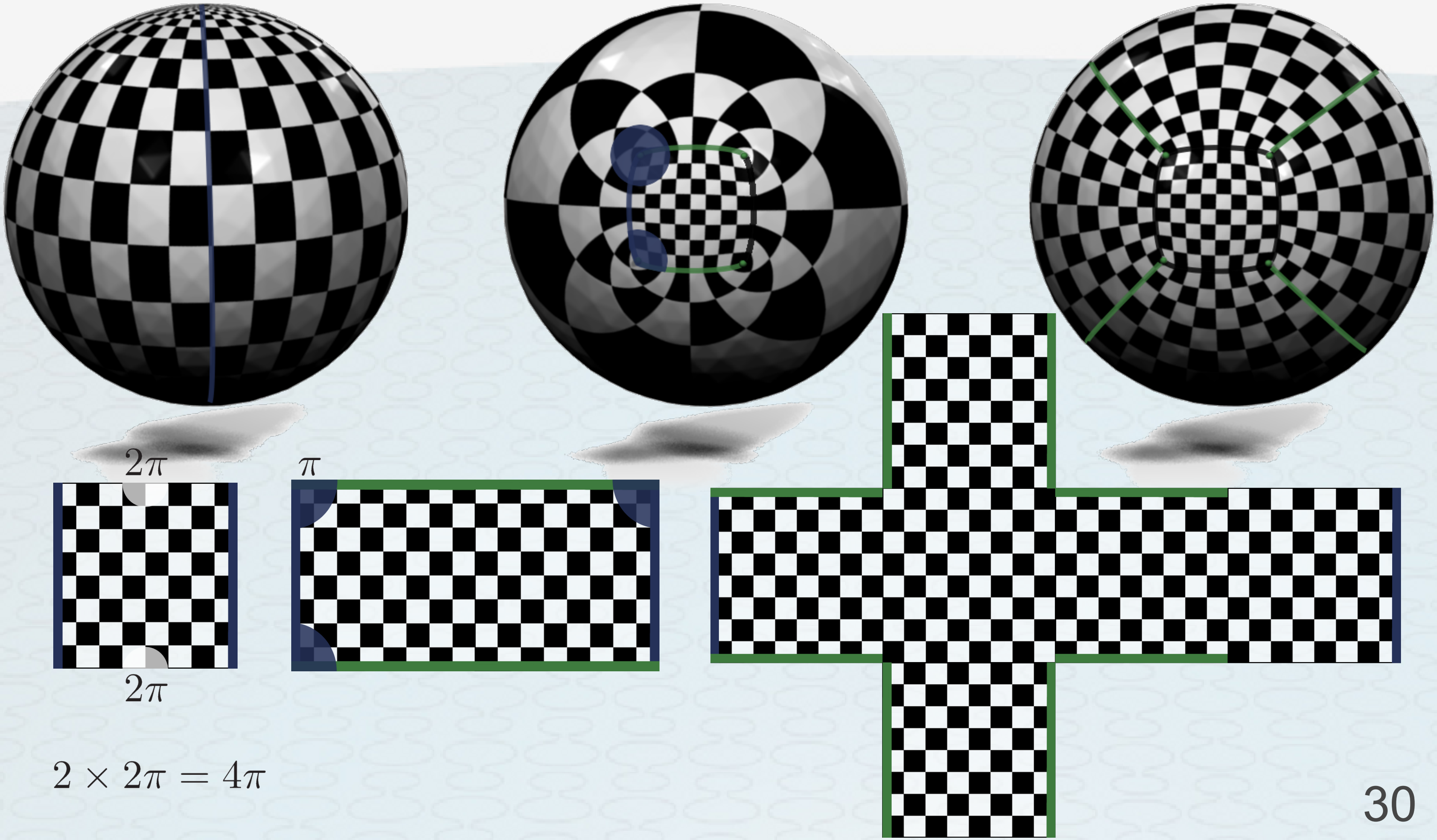
- How many *elastic* quadrilaterals do I need to wrap a sphere?



$2 \times 2\pi = 4\pi$

Gauss Bonnet theorem revisited (1/2)

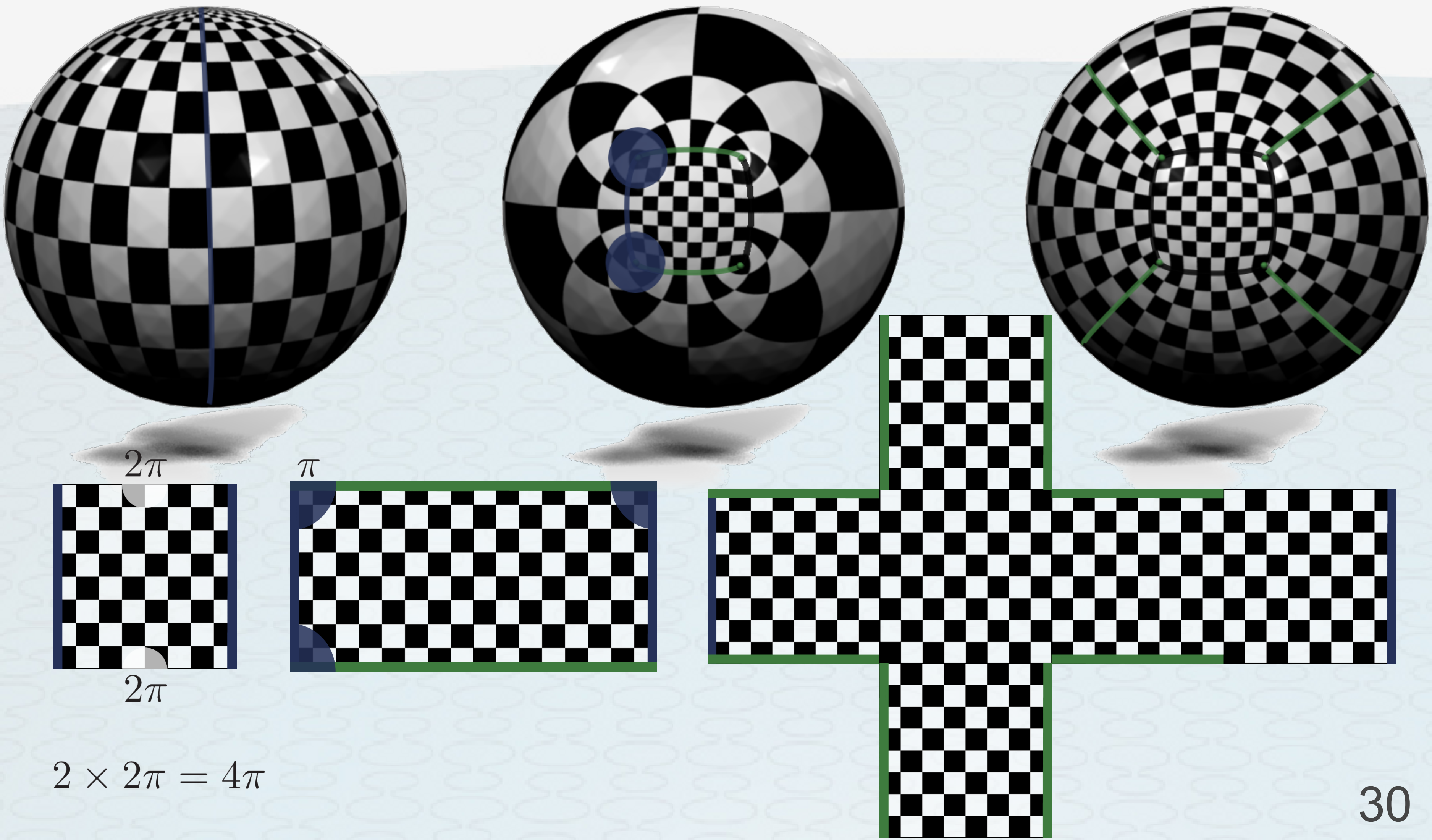
- How many *elastic* quadrilaterals do I need to wrap a sphere?



$2 \times 2\pi = 4\pi$

Gauss Bonnet theorem revisited (1/2)

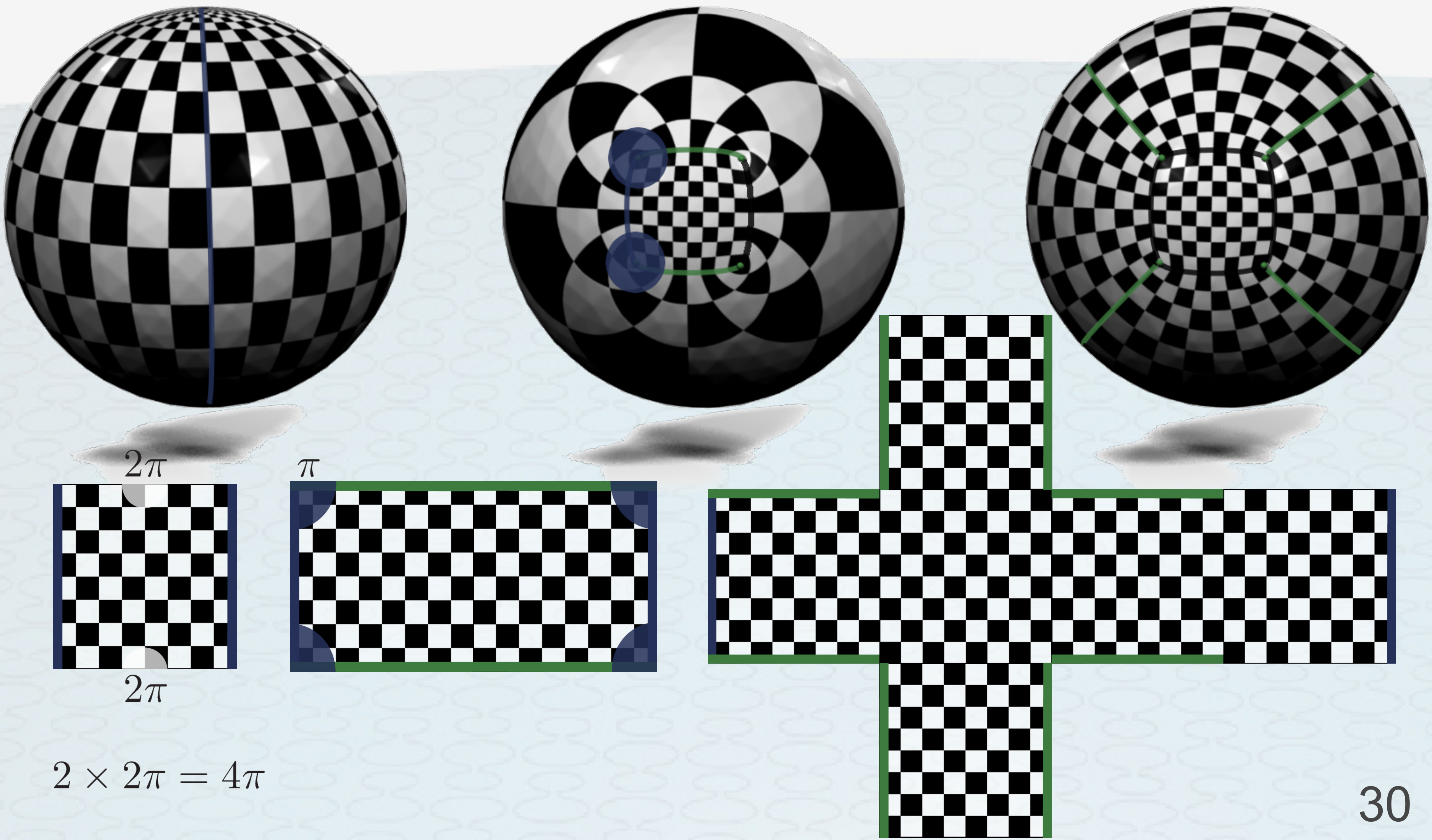
- How many *elastic* quadrilaterals do I need to wrap a sphere?



$$2 \times 2\pi = 4\pi$$

Gauss Bonnet theorem revisited (1/2)

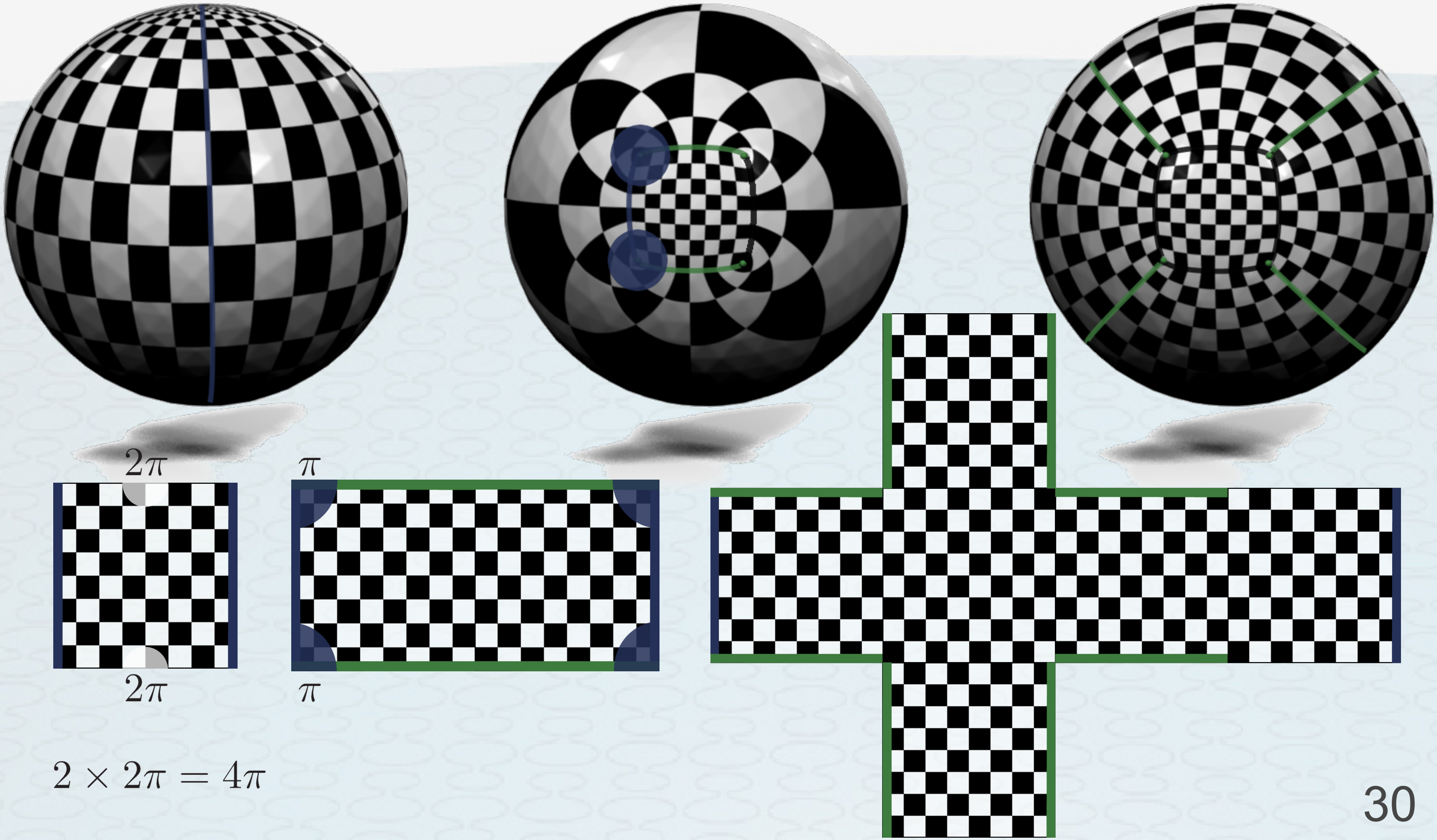
- How many *elastic* quadrilaterals do I need to wrap a sphere?



$2 \times 2\pi = 4\pi$

Gauss Bonnet theorem revisited (1/2)

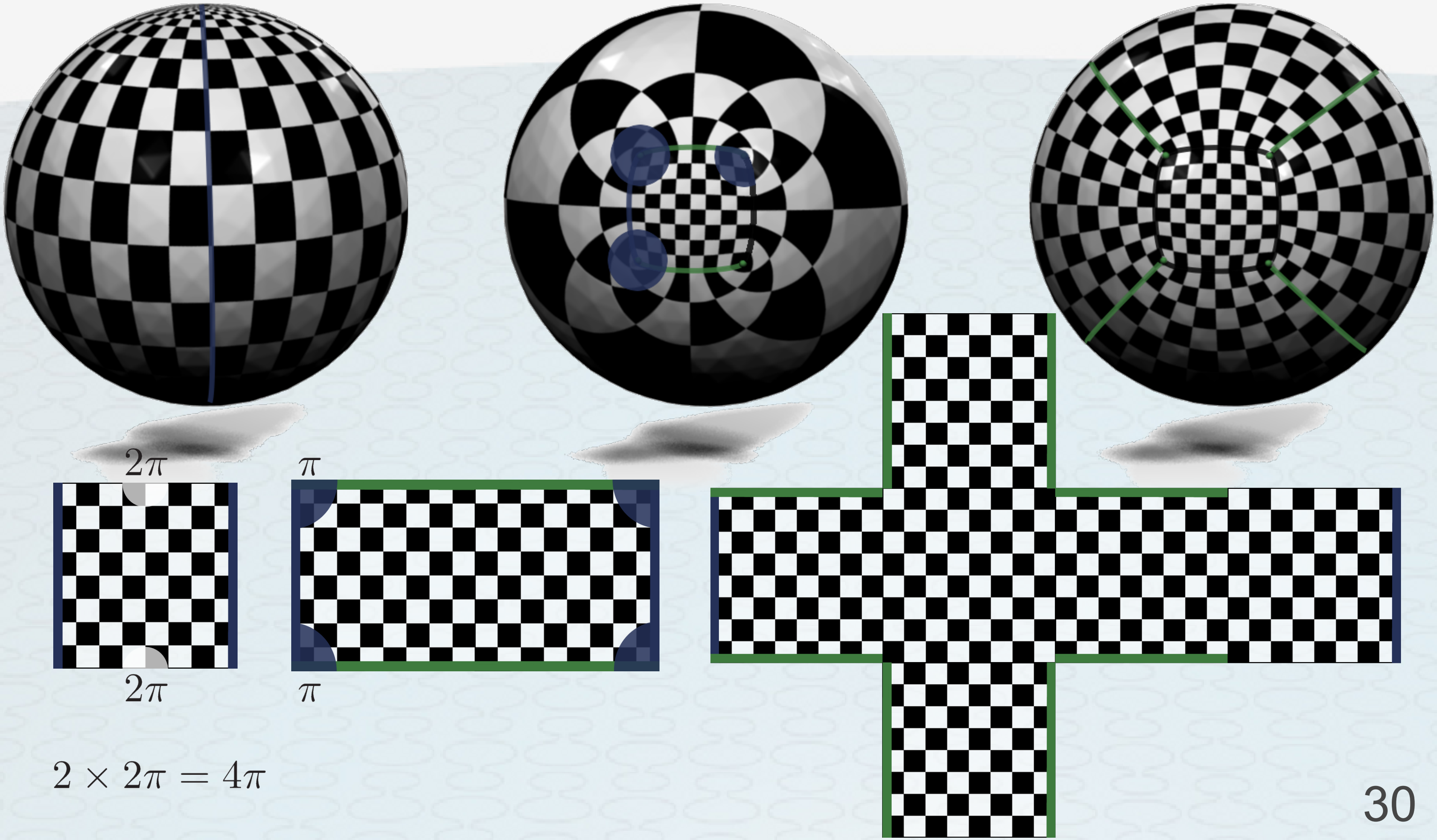
- How many *elastic* quadrilaterals do I need to wrap a sphere?



$$2 \times 2\pi = 4\pi$$

Gauss Bonnet theorem revisited (1/2)

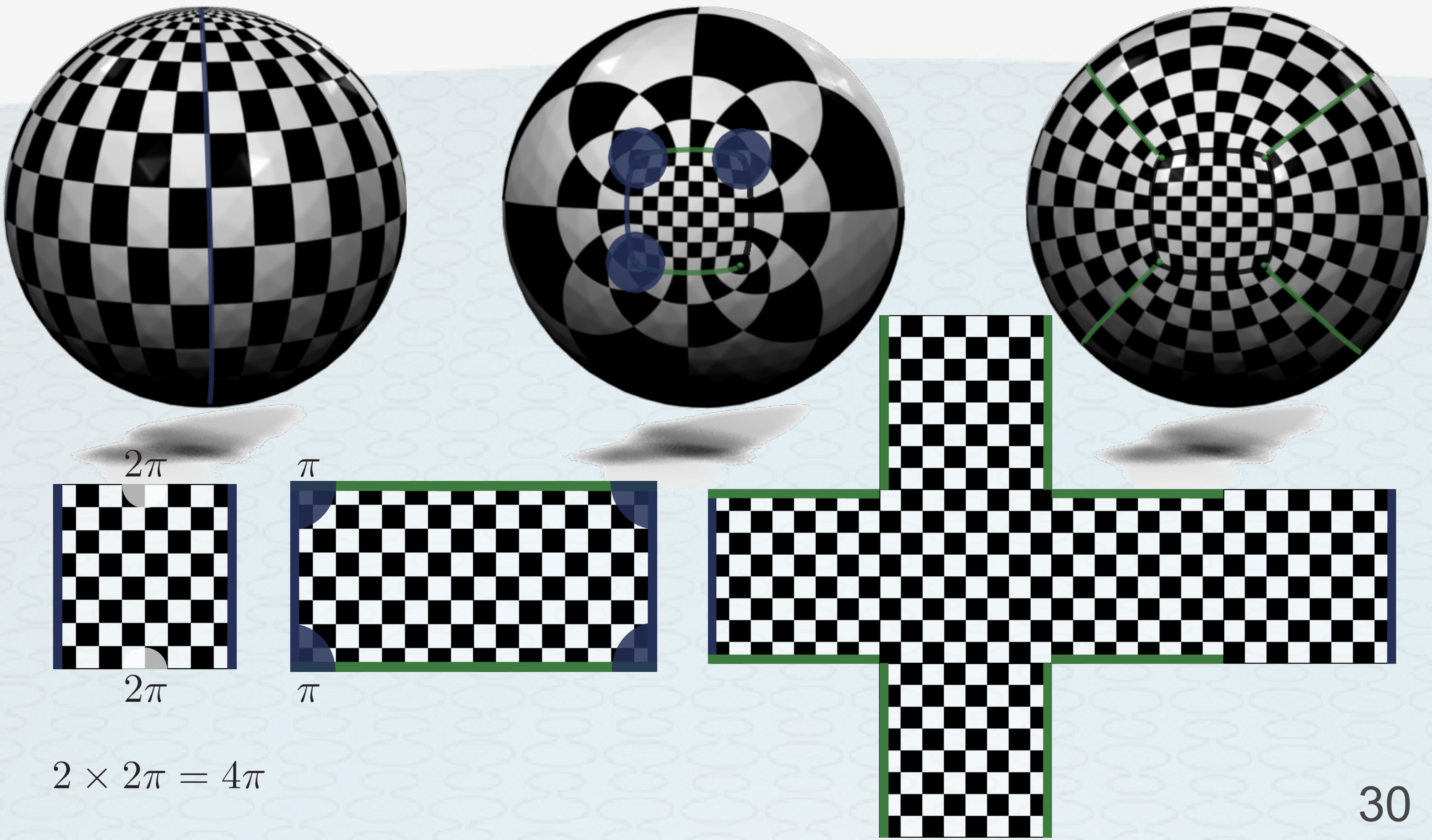
- How many *elastic* quadrilaterals do I need to wrap a sphere?



$2 \times 2\pi = 4\pi$

Gauss Bonnet theorem revisited (1/2)

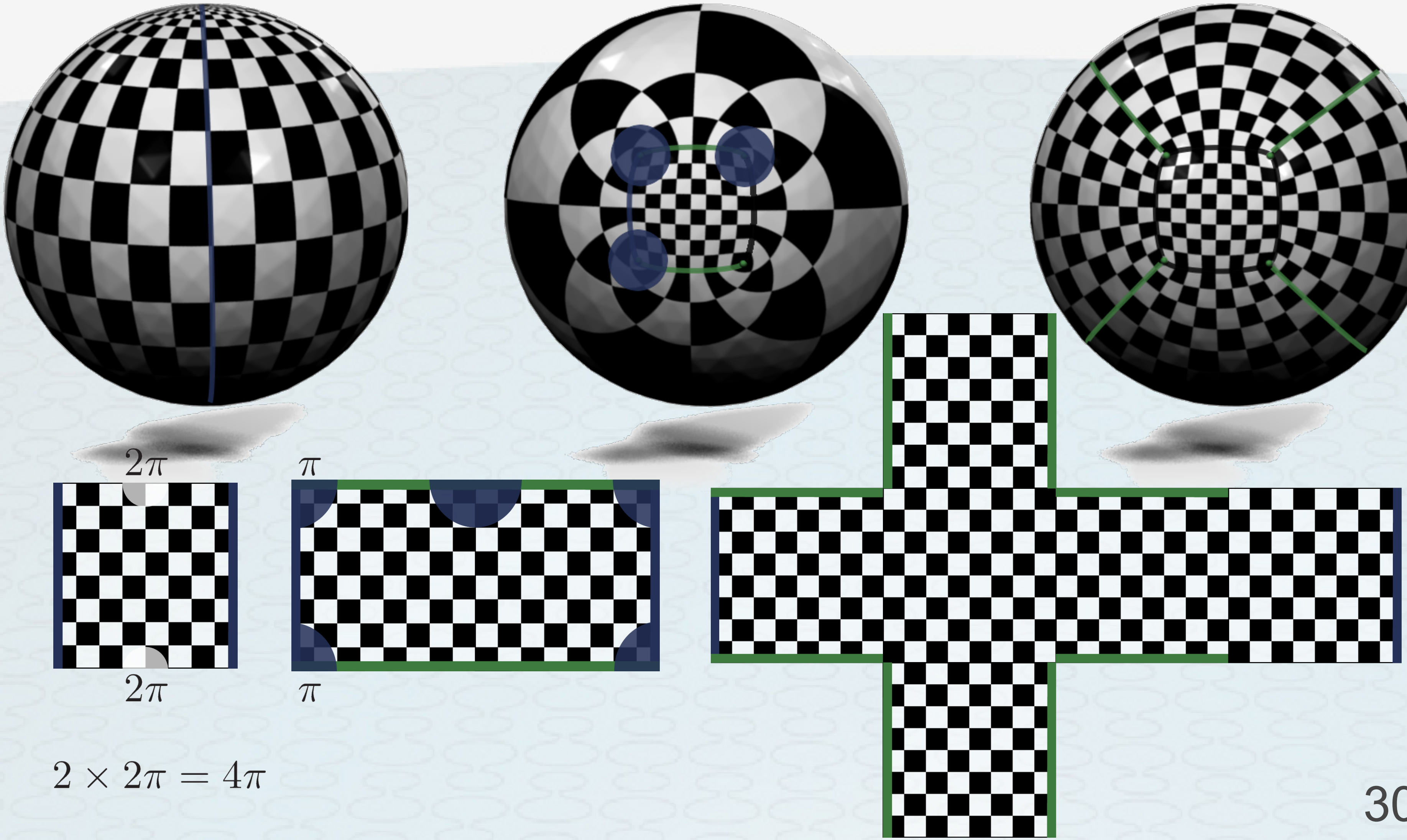
- How many *elastic* quadrilaterals do I need to wrap a sphere?



$2 \times 2\pi = 4\pi$

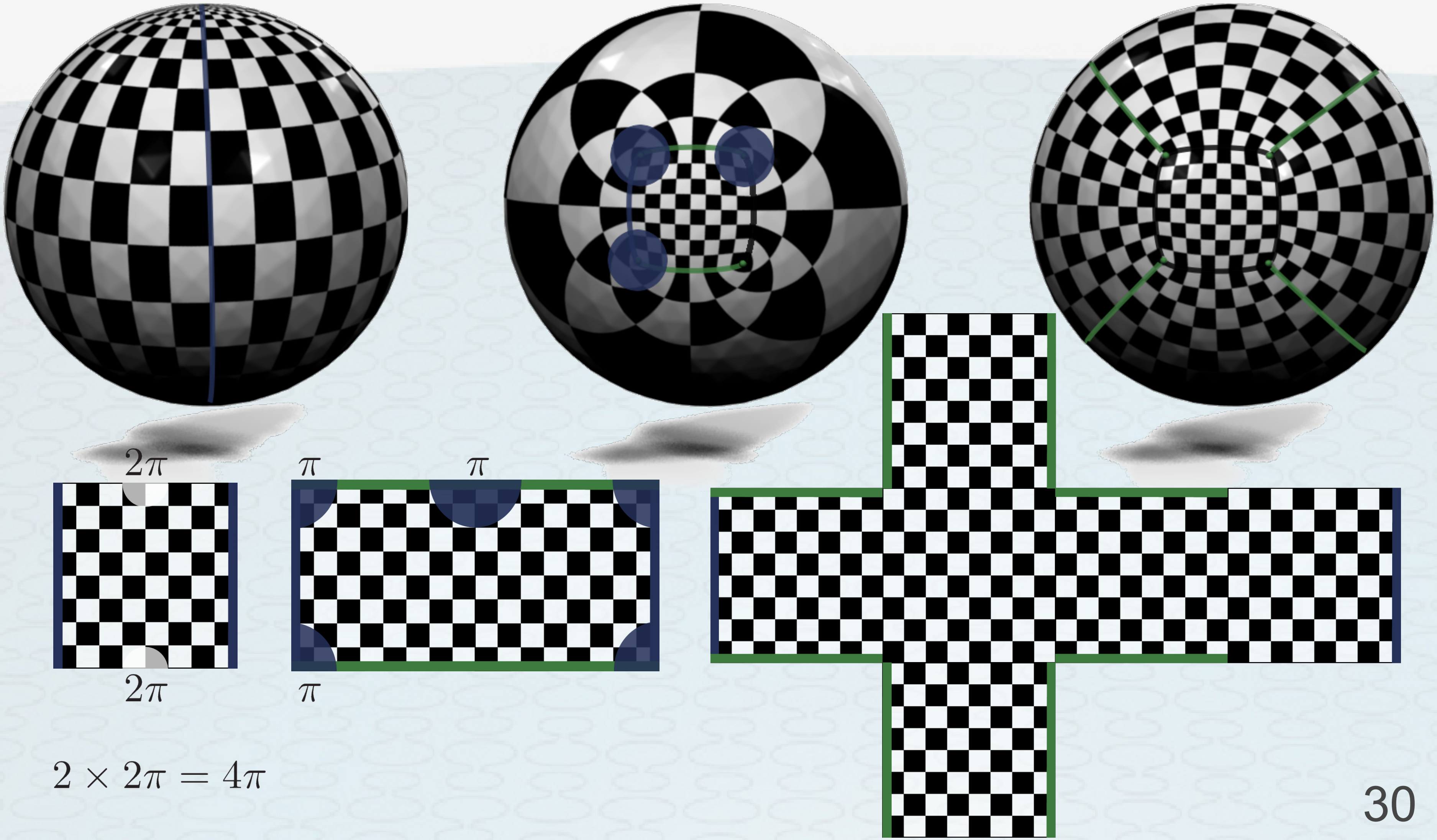
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



Gauss Bonnet theorem revisited (1/2)

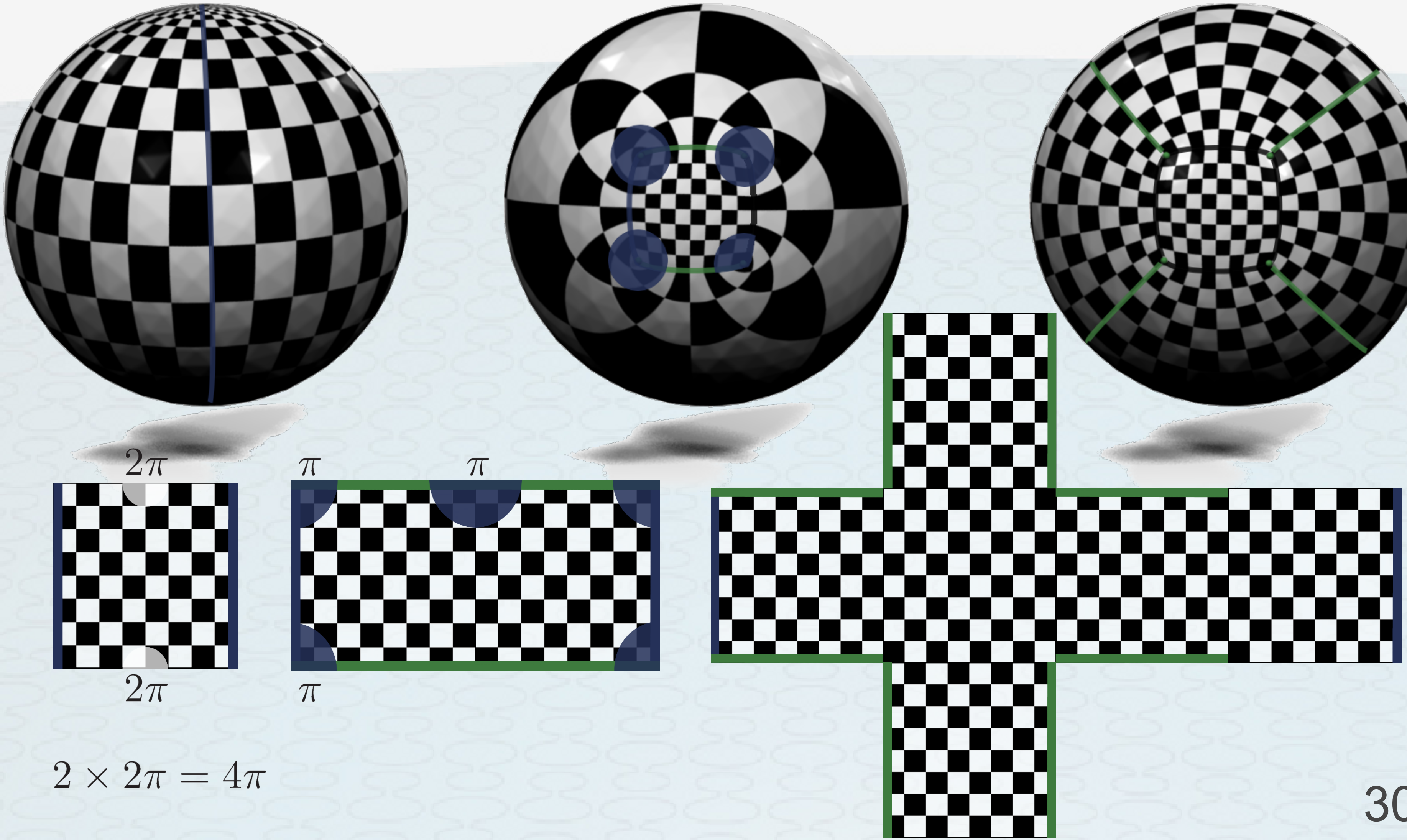
- How many *elastic* quadrilaterals do I need to wrap a sphere?



$2 \times 2\pi = 4\pi$

Gauss Bonnet theorem revisited (1/2)

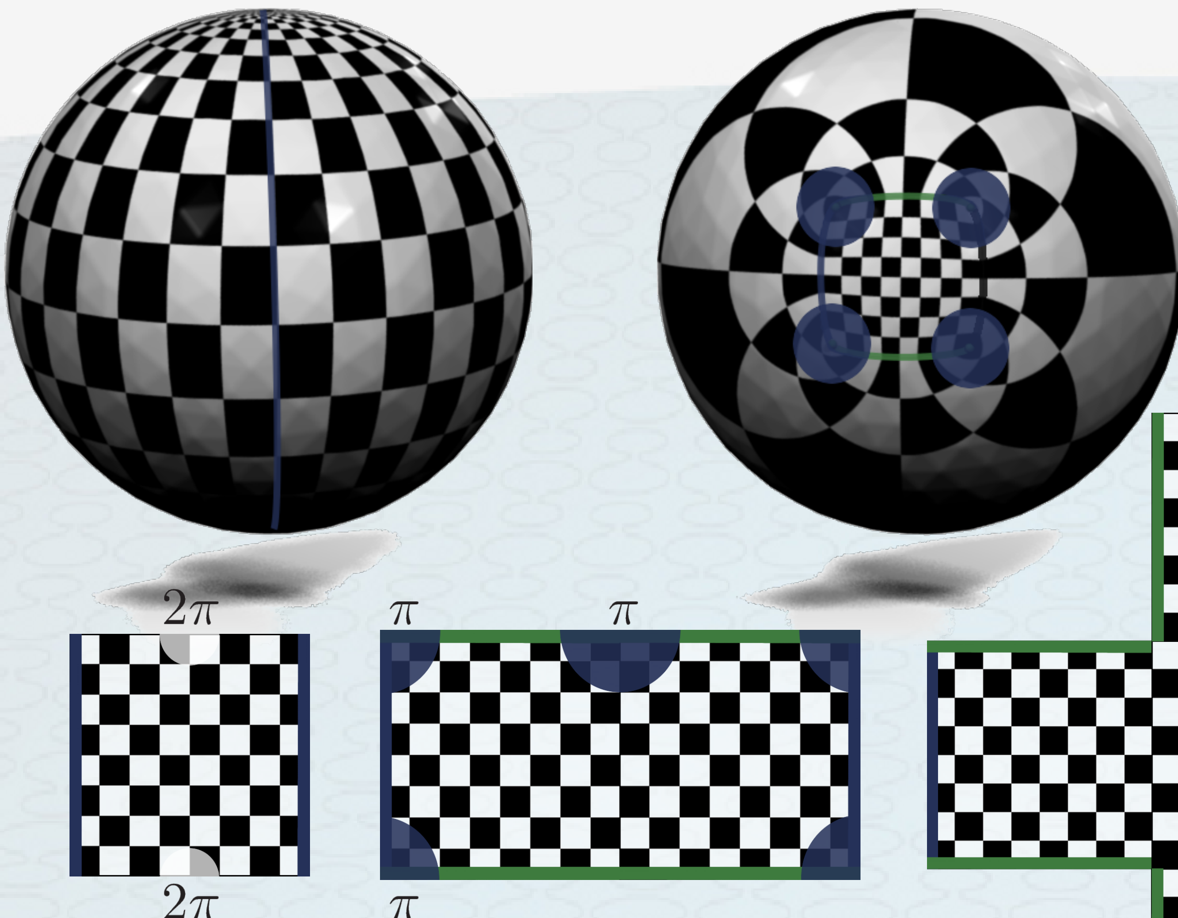
- How many *elastic* quadrilaterals do I need to wrap a sphere?

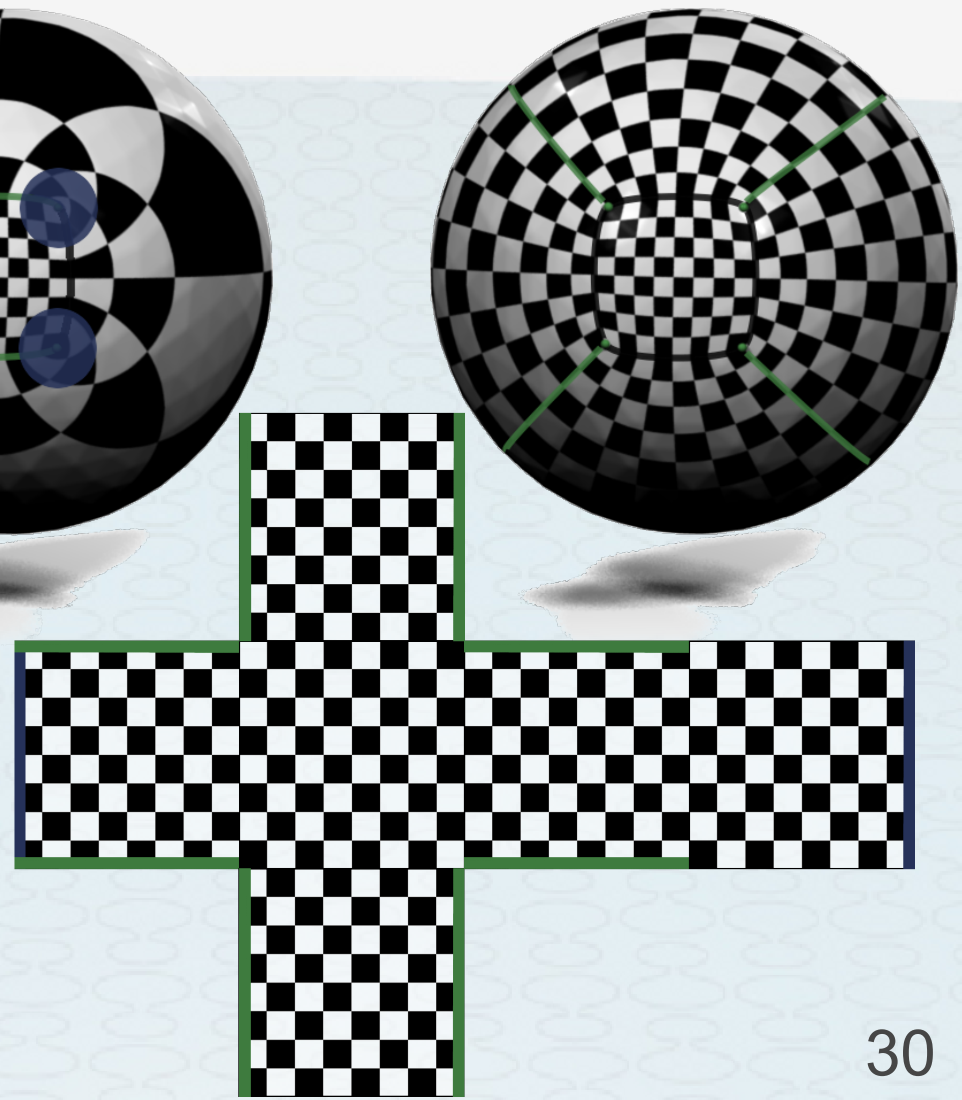


$2 \times 2\pi = 4\pi$

Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?

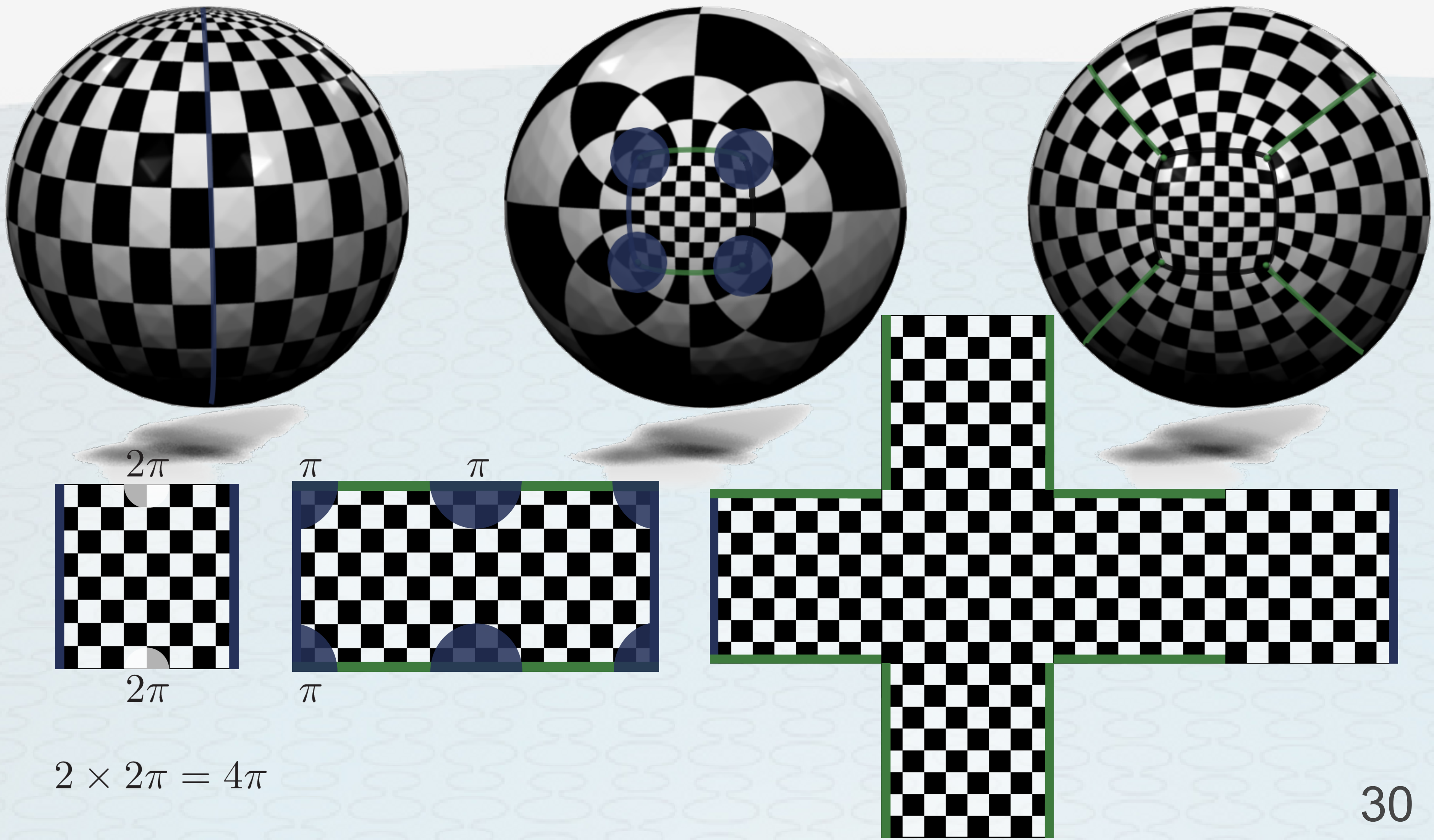




$2 \times 2\pi = 4\pi$

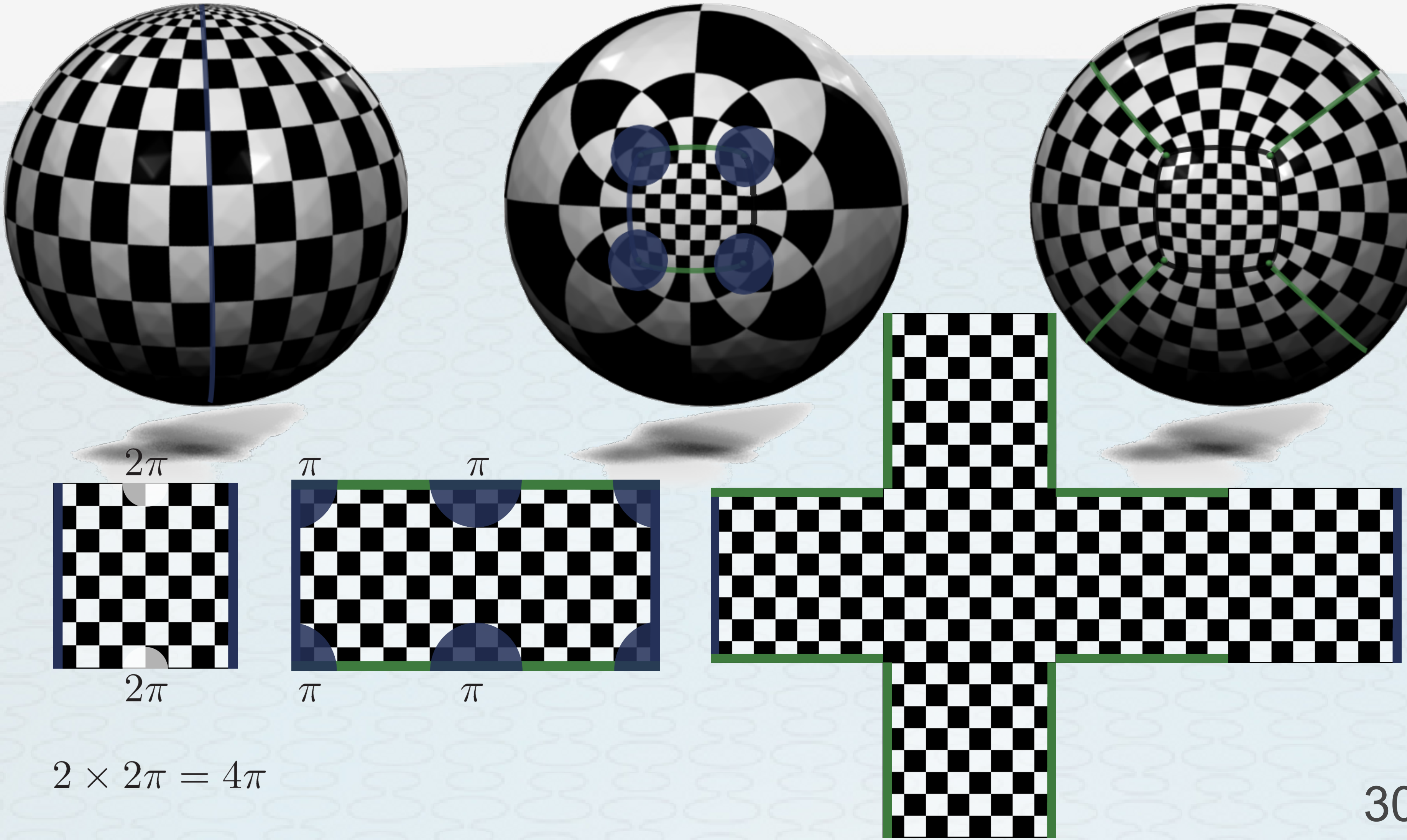
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



Gauss Bonnet theorem revisited (1/2)

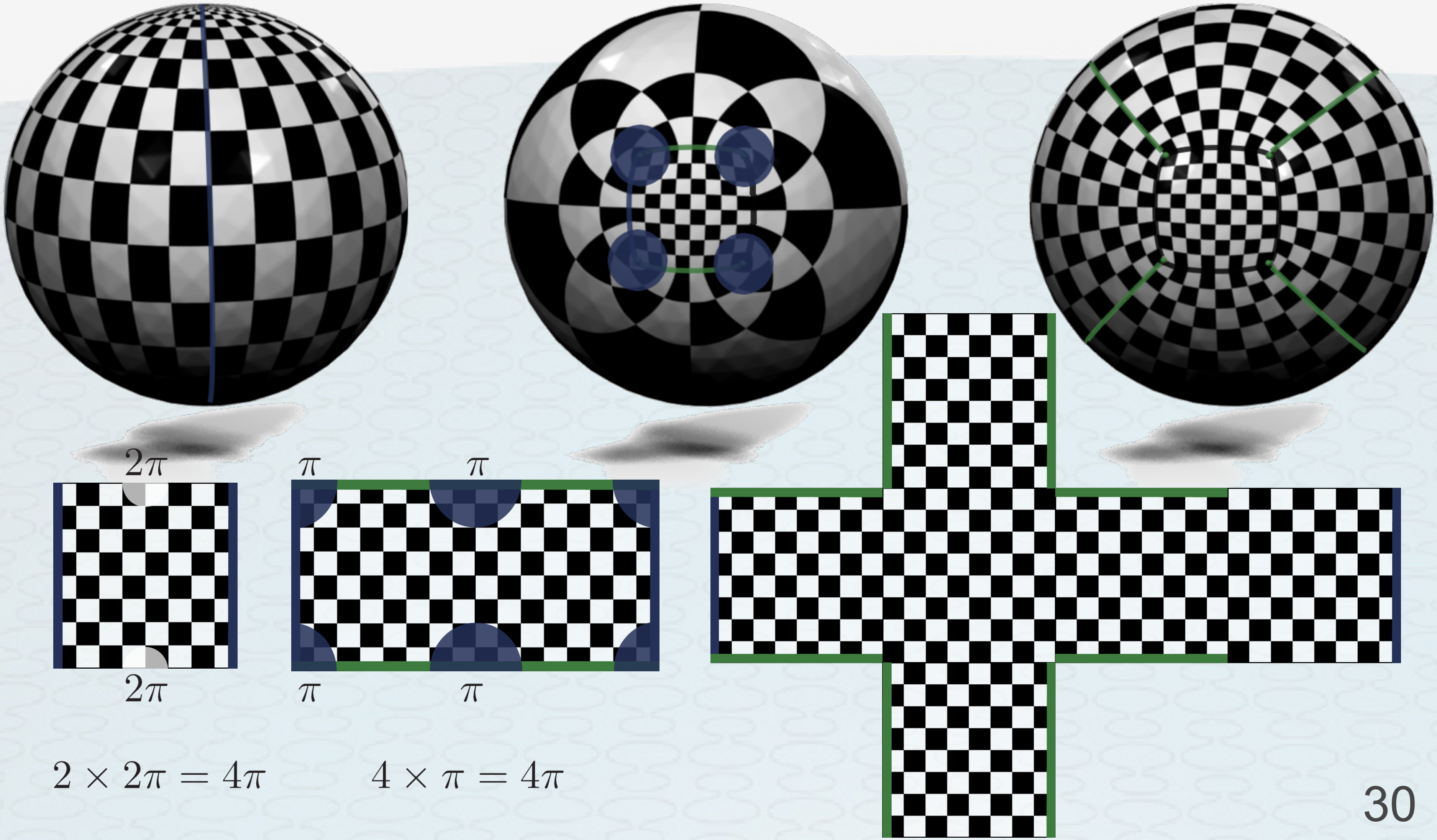
- How many *elastic* quadrilaterals do I need to wrap a sphere?



$2 \times 2\pi = 4\pi$

Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?

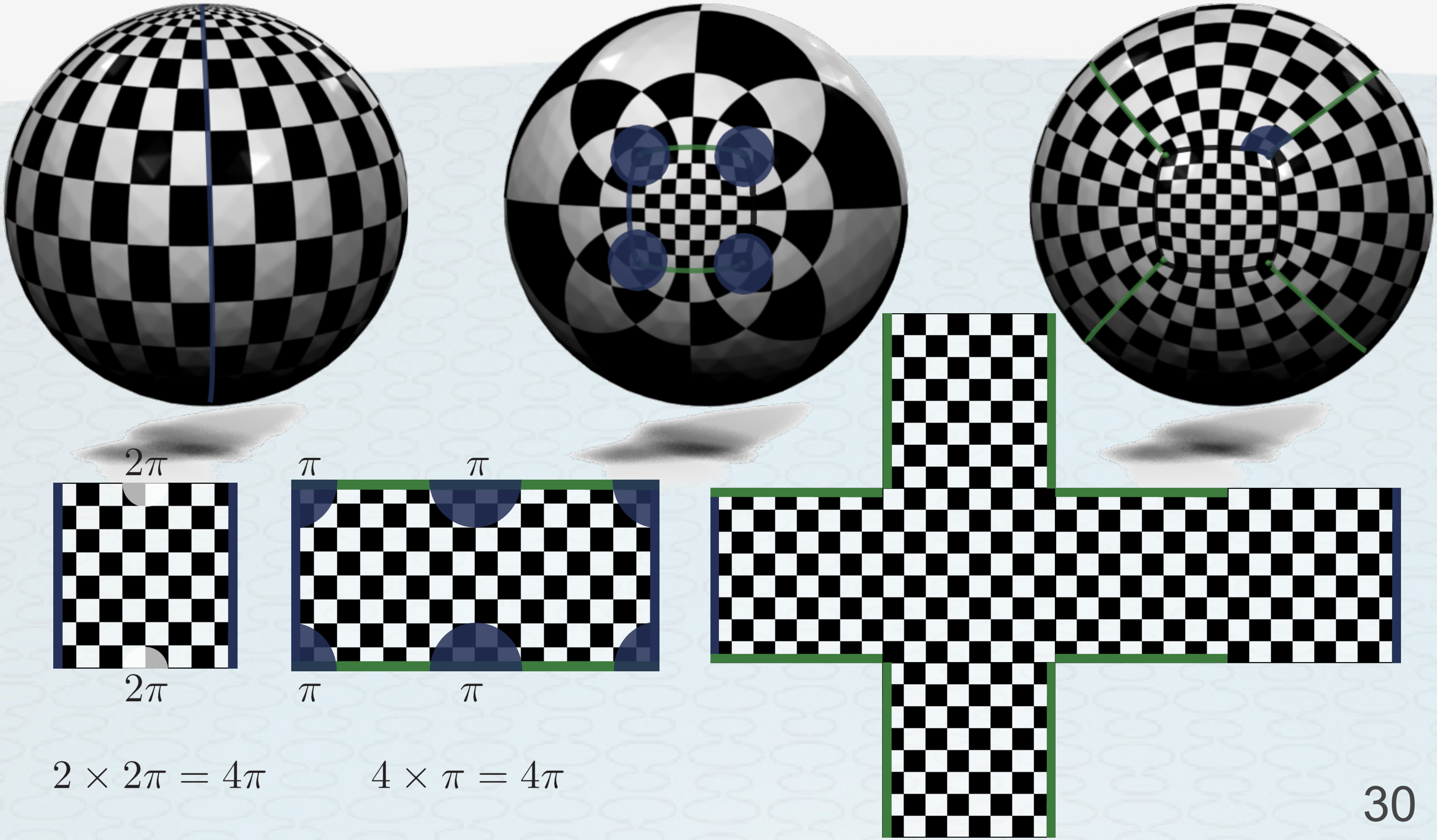


$2 \times 2\pi = 4\pi$

$4 \times \pi = 4\pi$

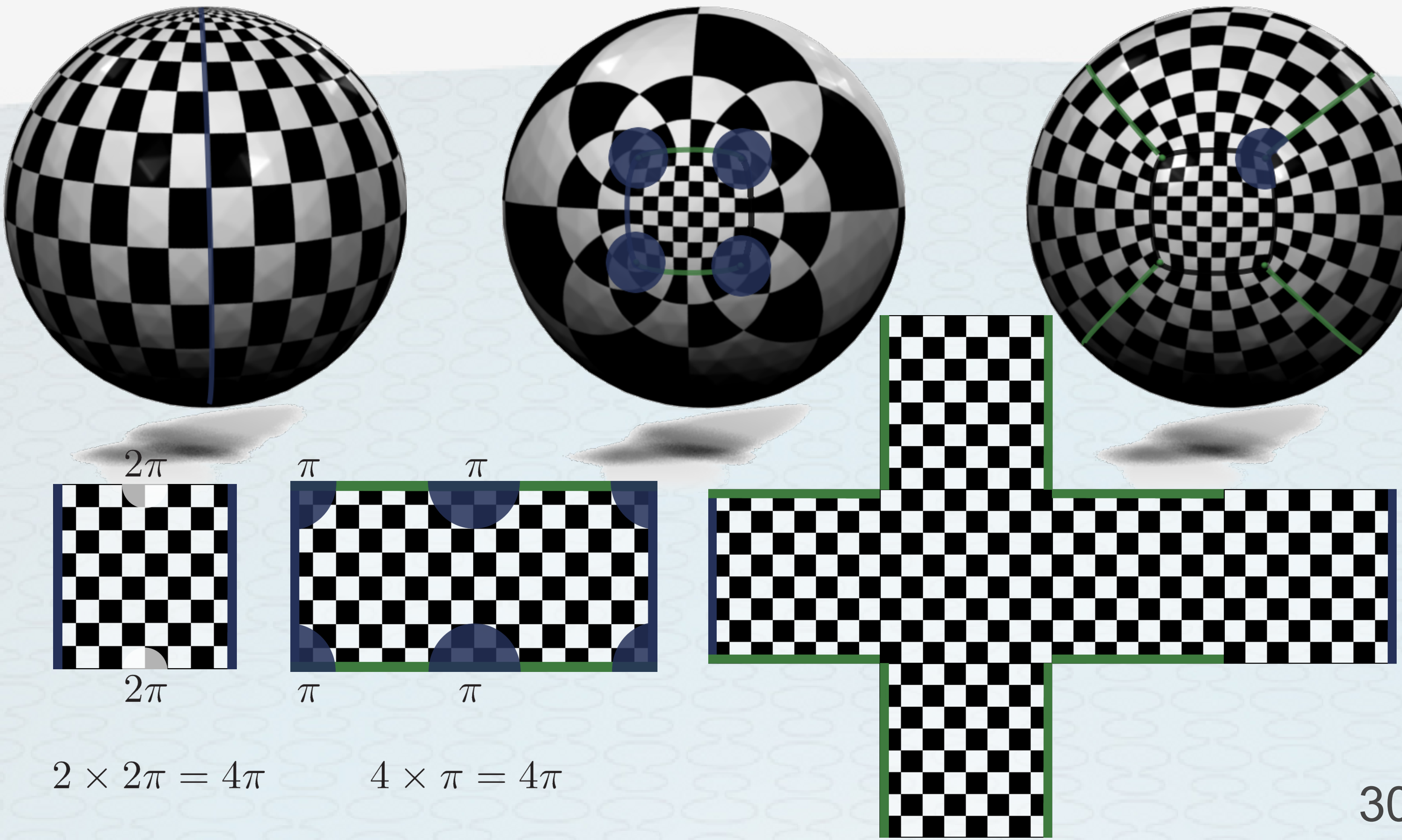
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



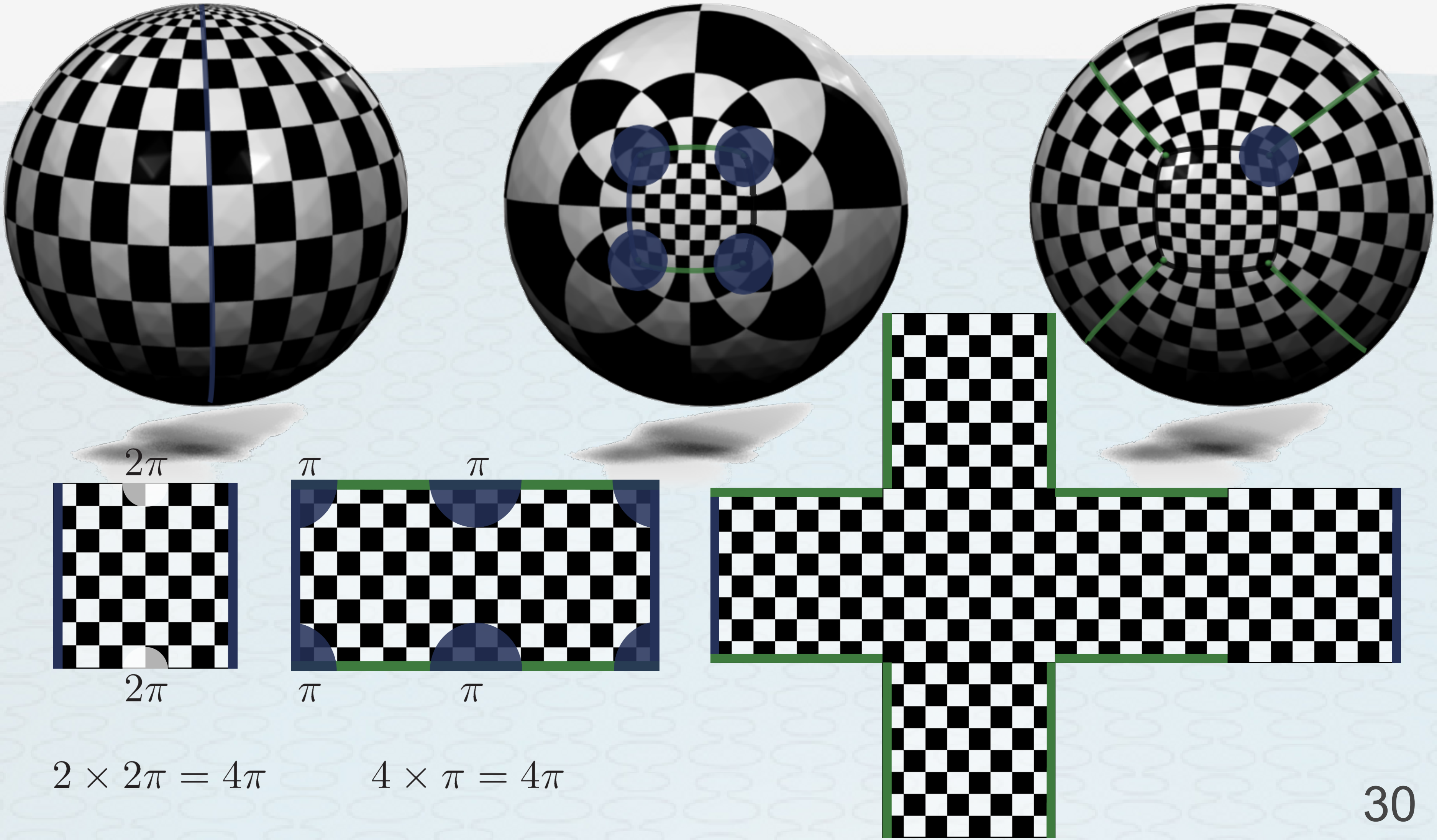
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



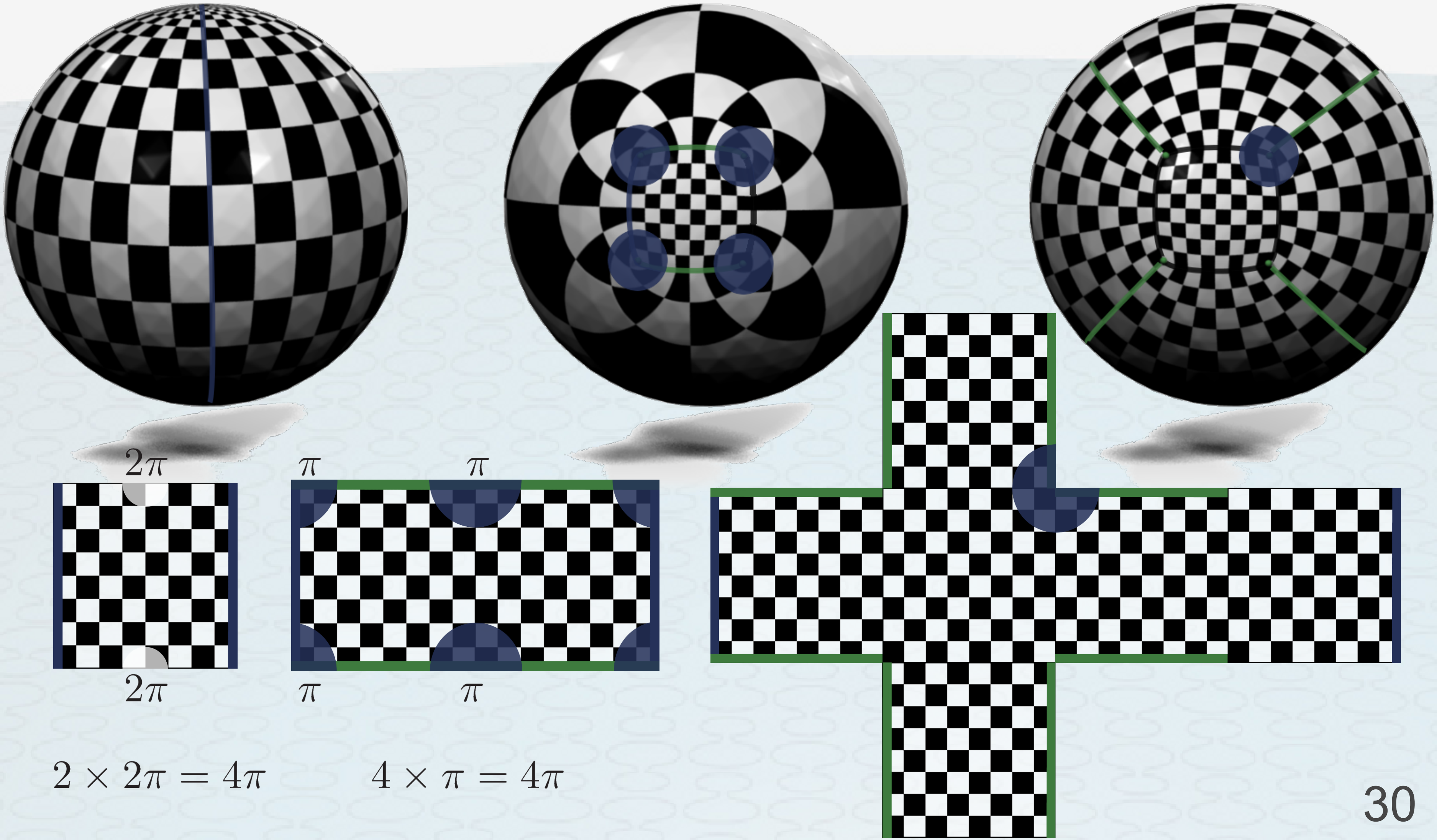
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?

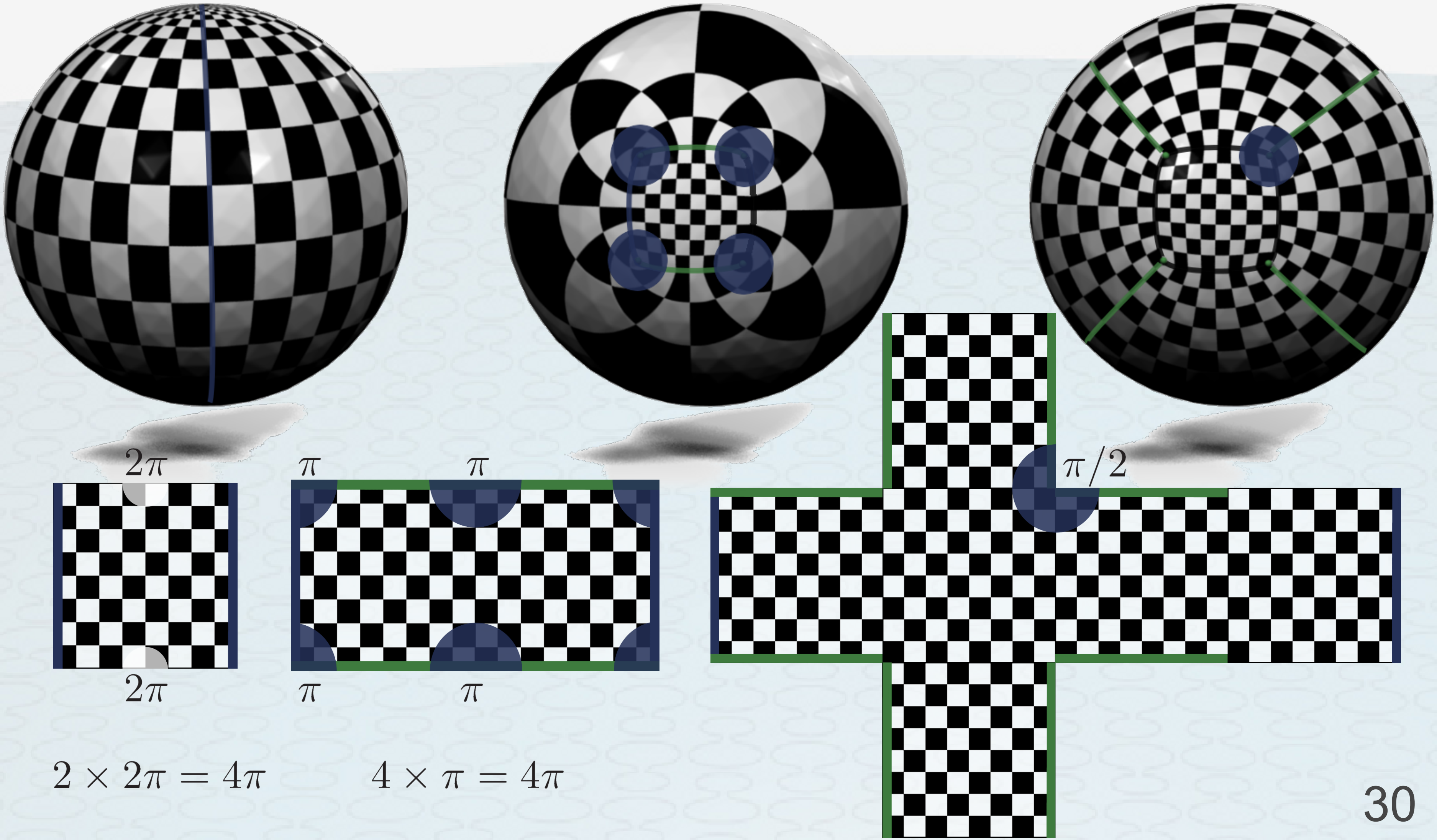


$2 \times 2\pi = 4\pi$

$4 \times \pi = 4\pi$

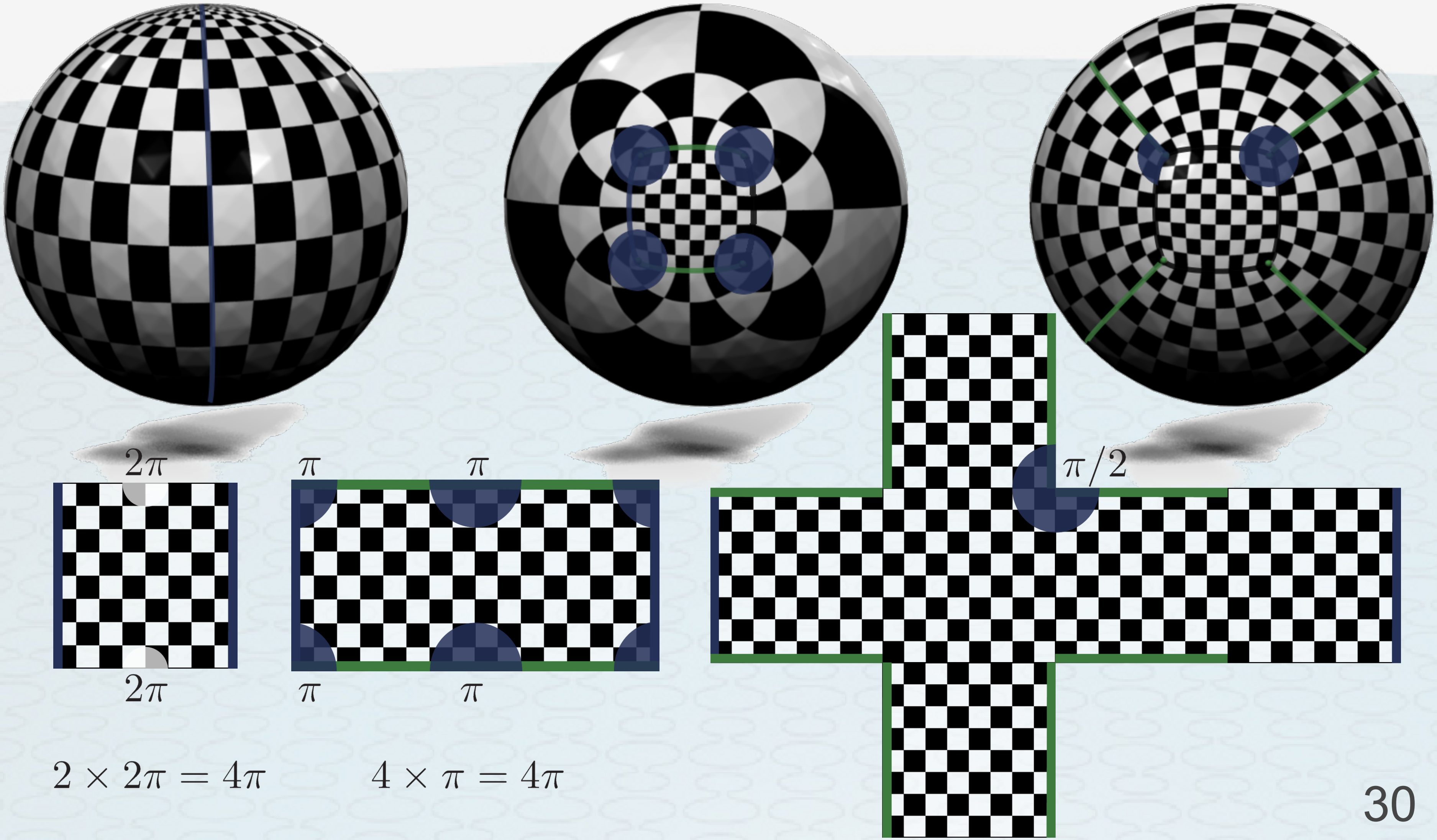
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



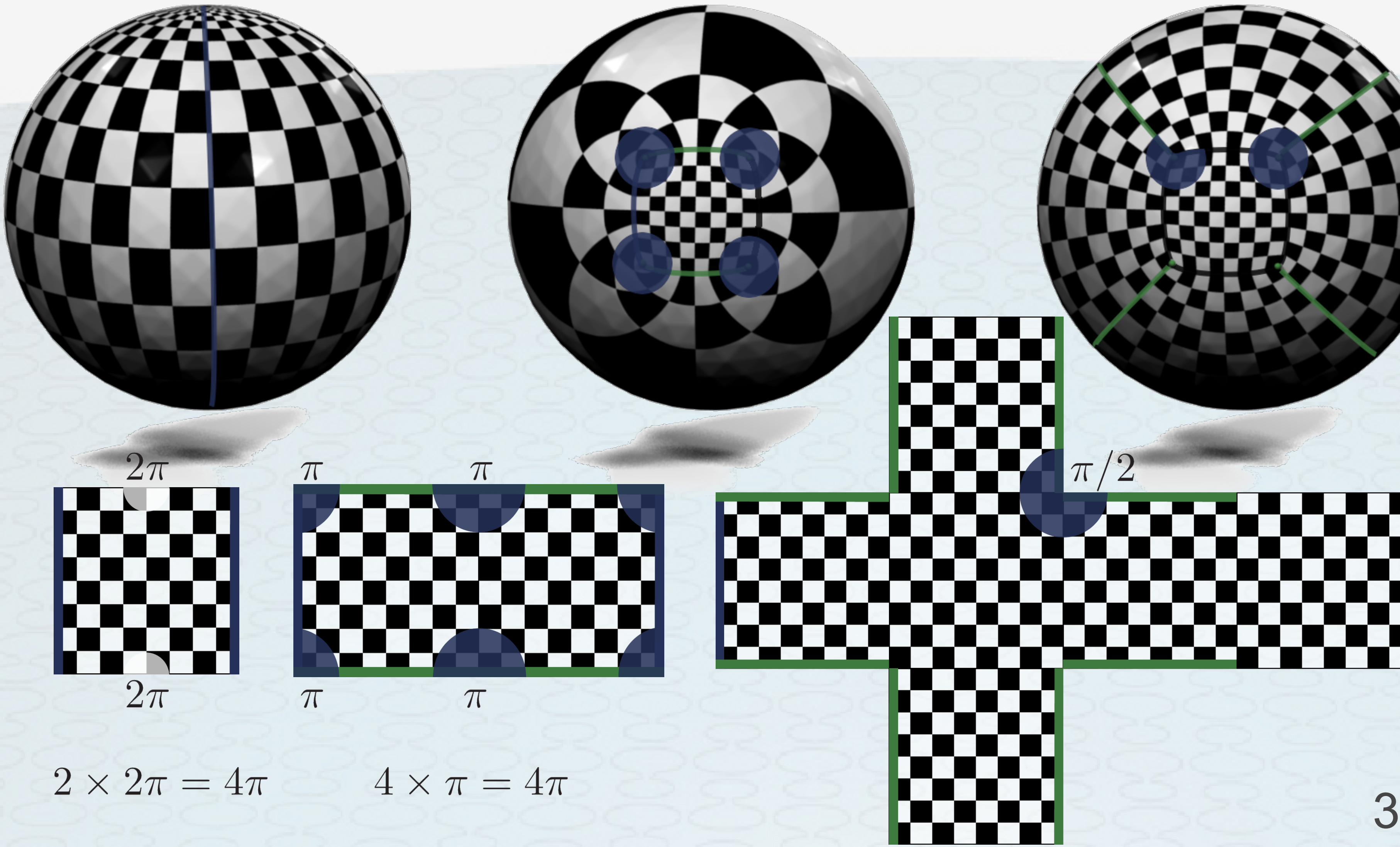
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?

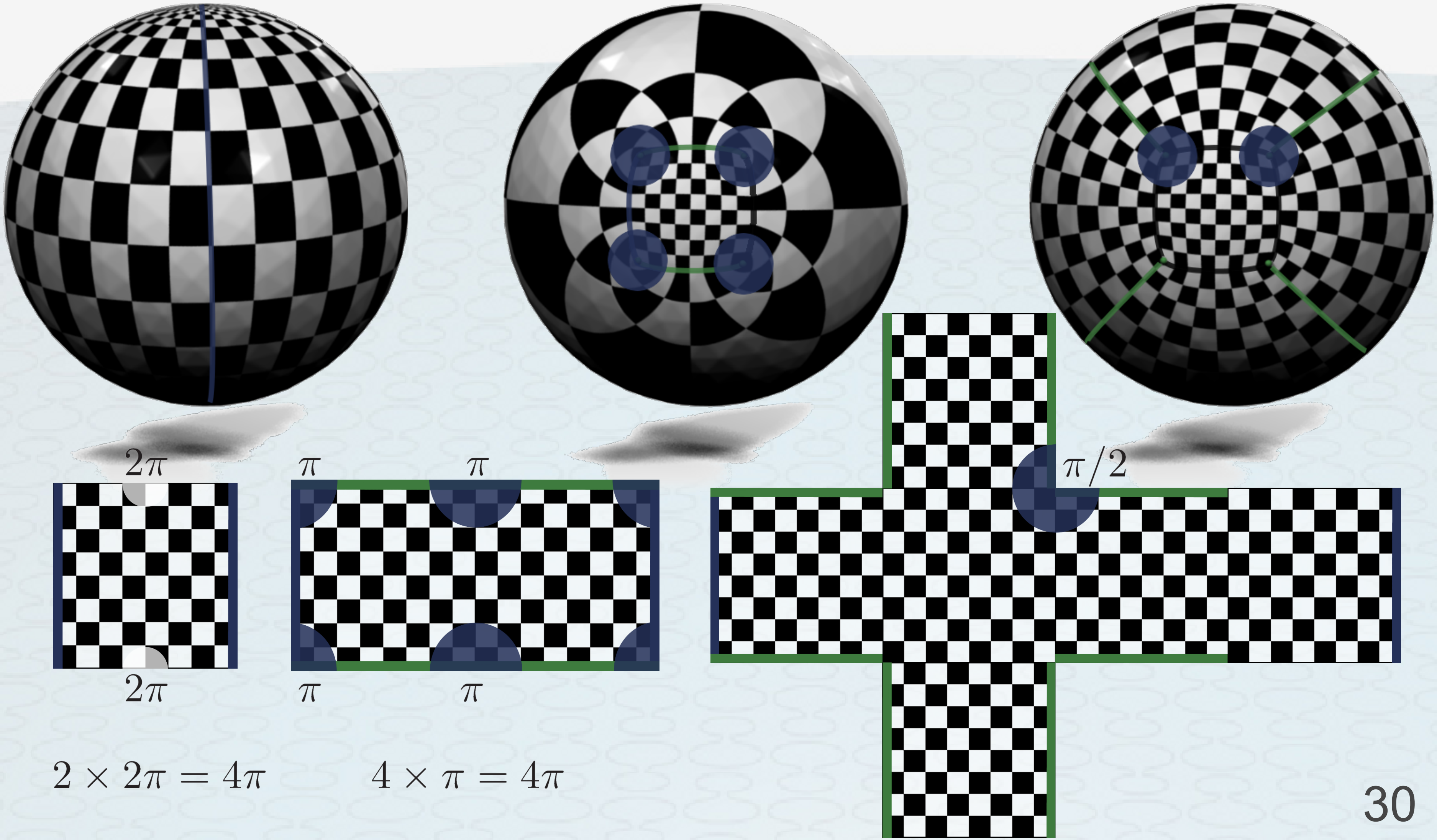


$2 \times 2\pi = 4\pi$

$4 \times \pi = 4\pi$

Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?

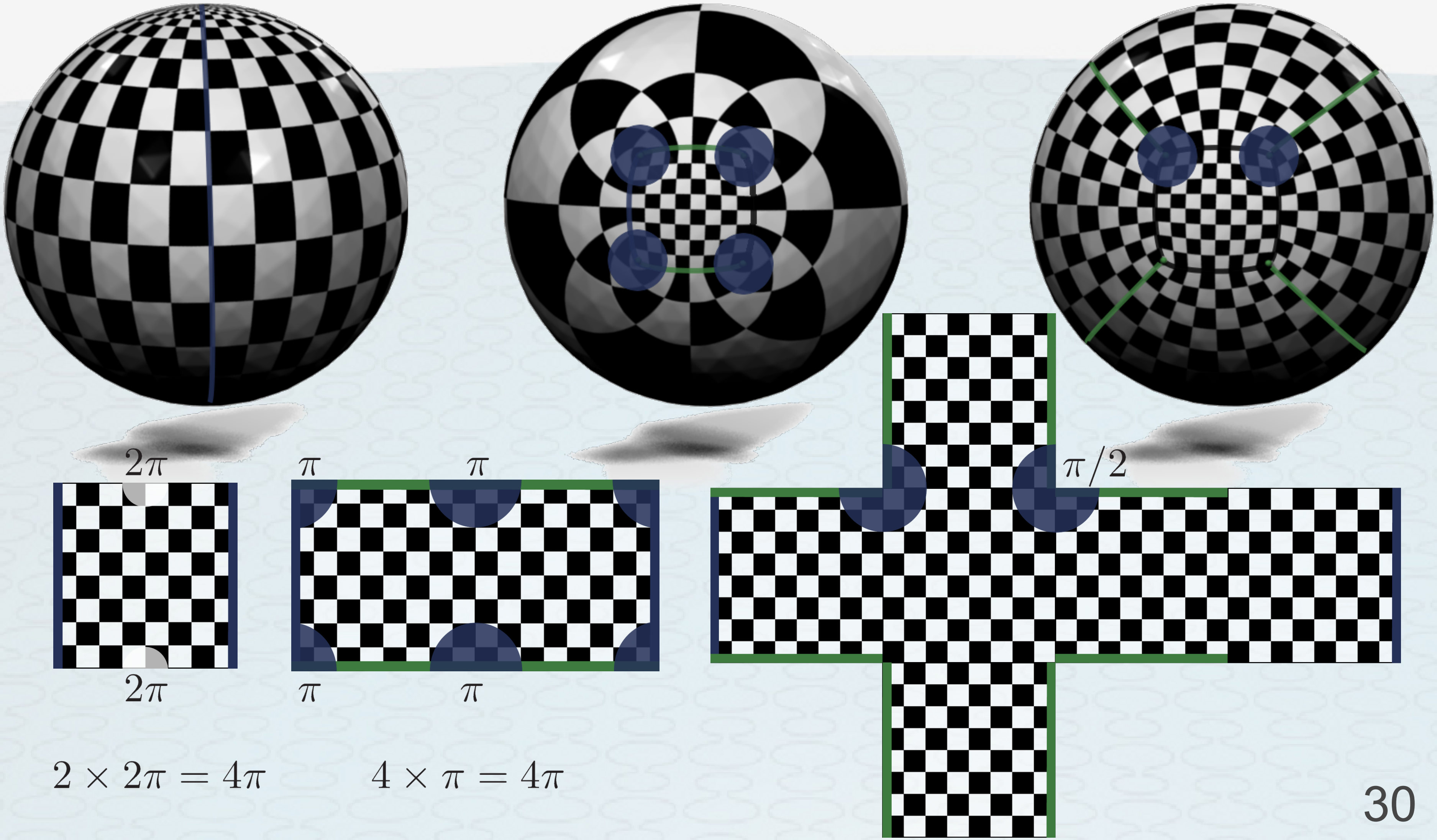


$2 \times 2\pi = 4\pi$

$4 \times \pi = 4\pi$

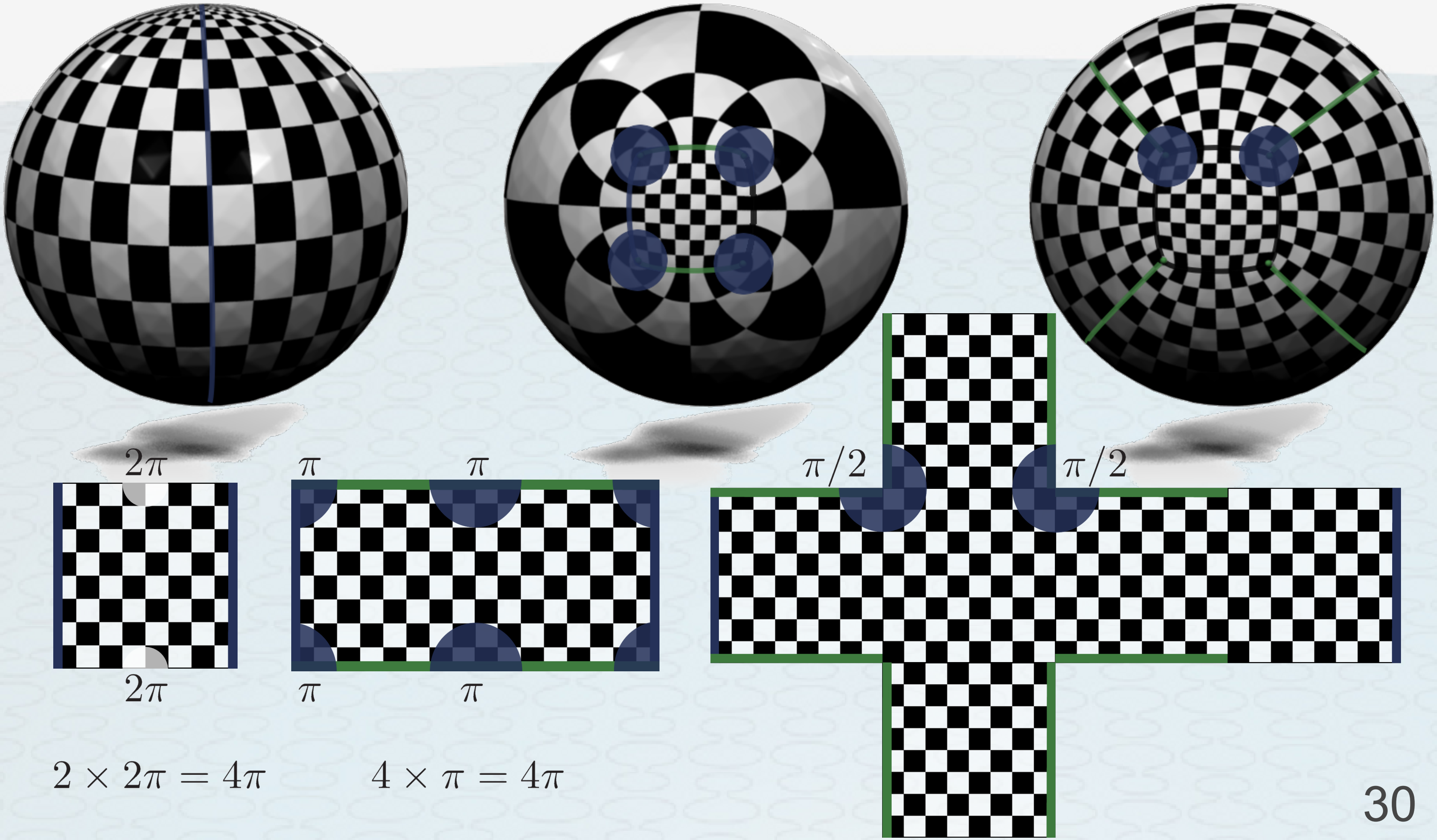
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



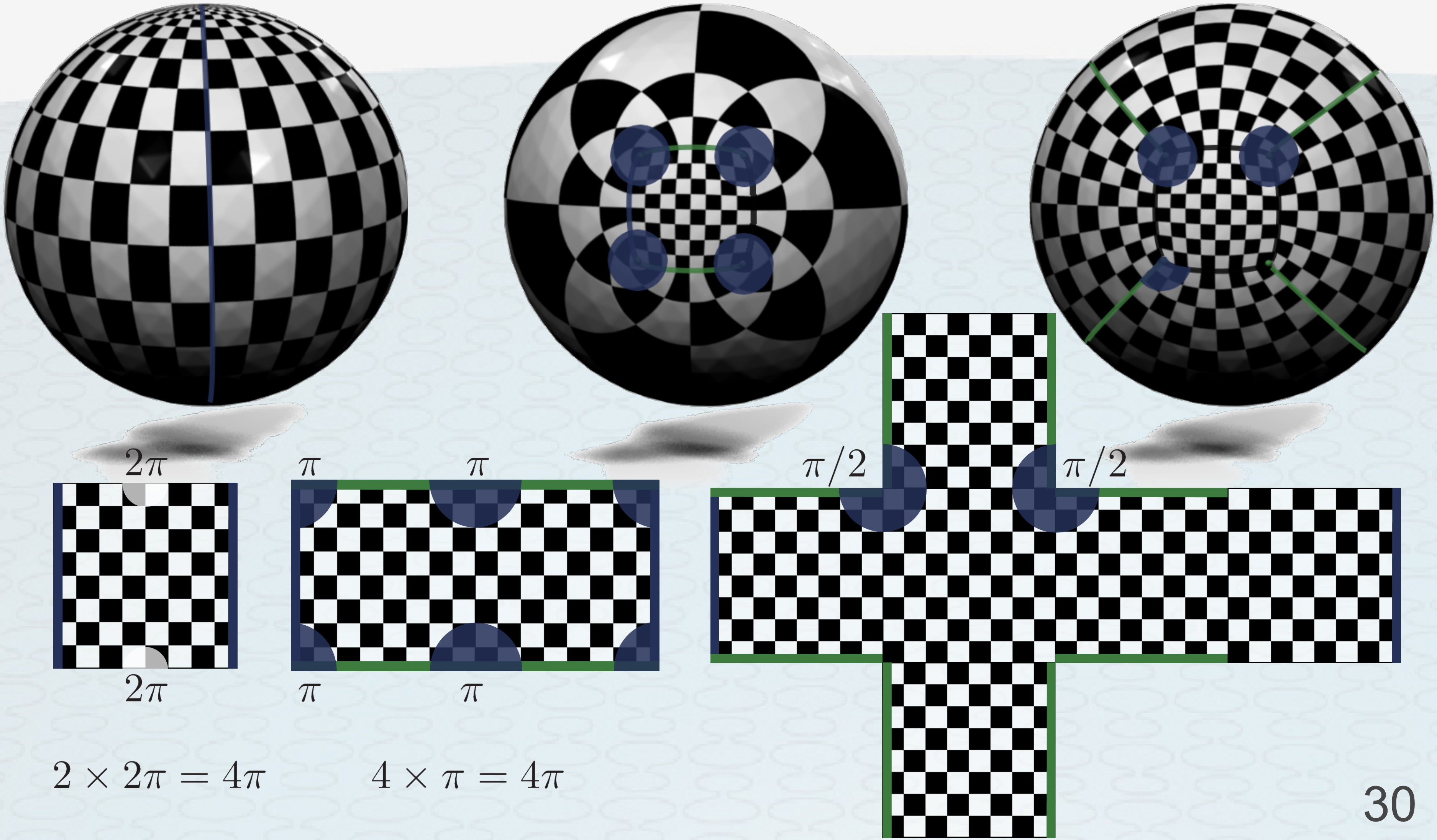
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?

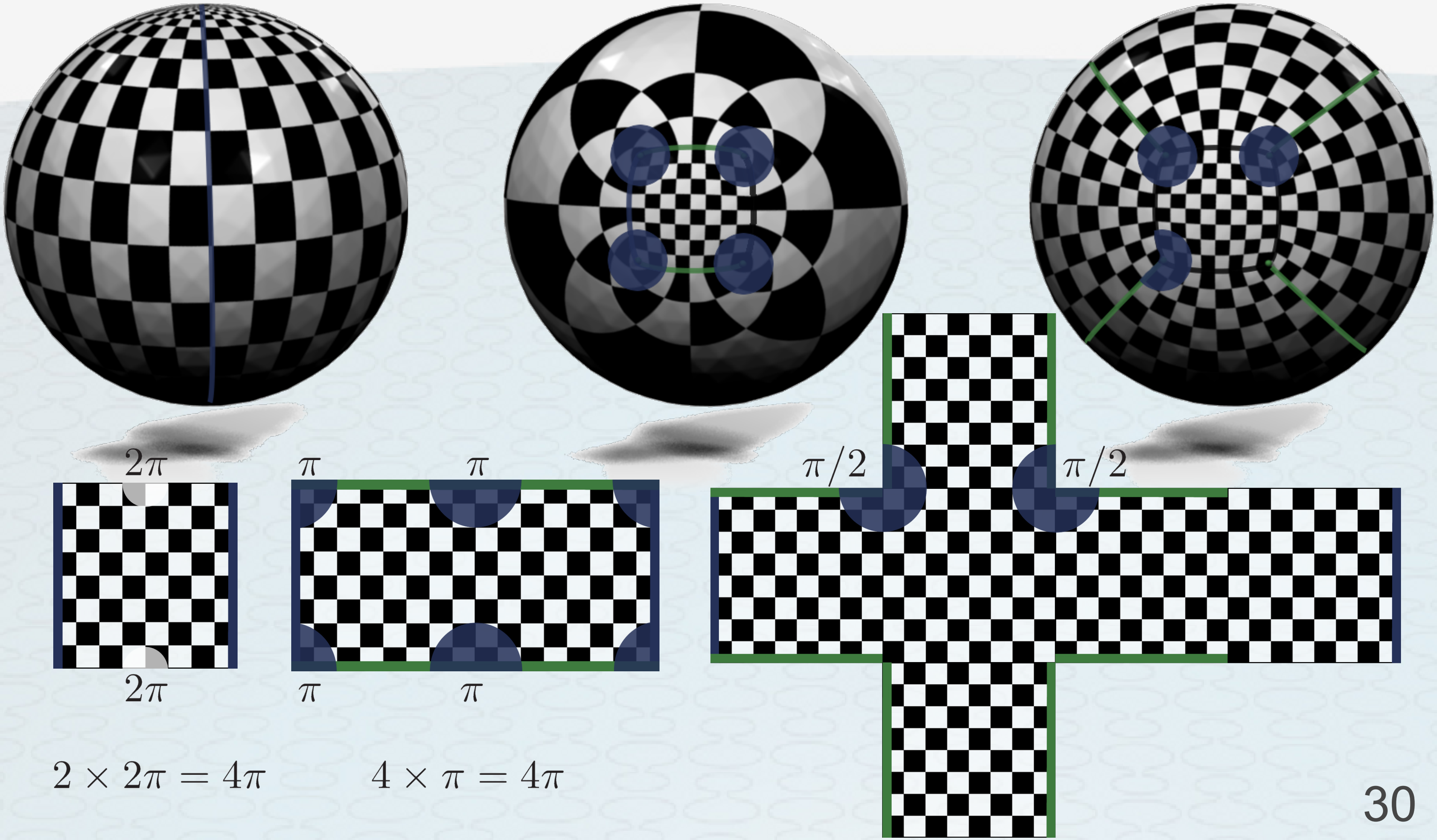


$2 \times 2\pi = 4\pi$

$4 \times \pi = 4\pi$

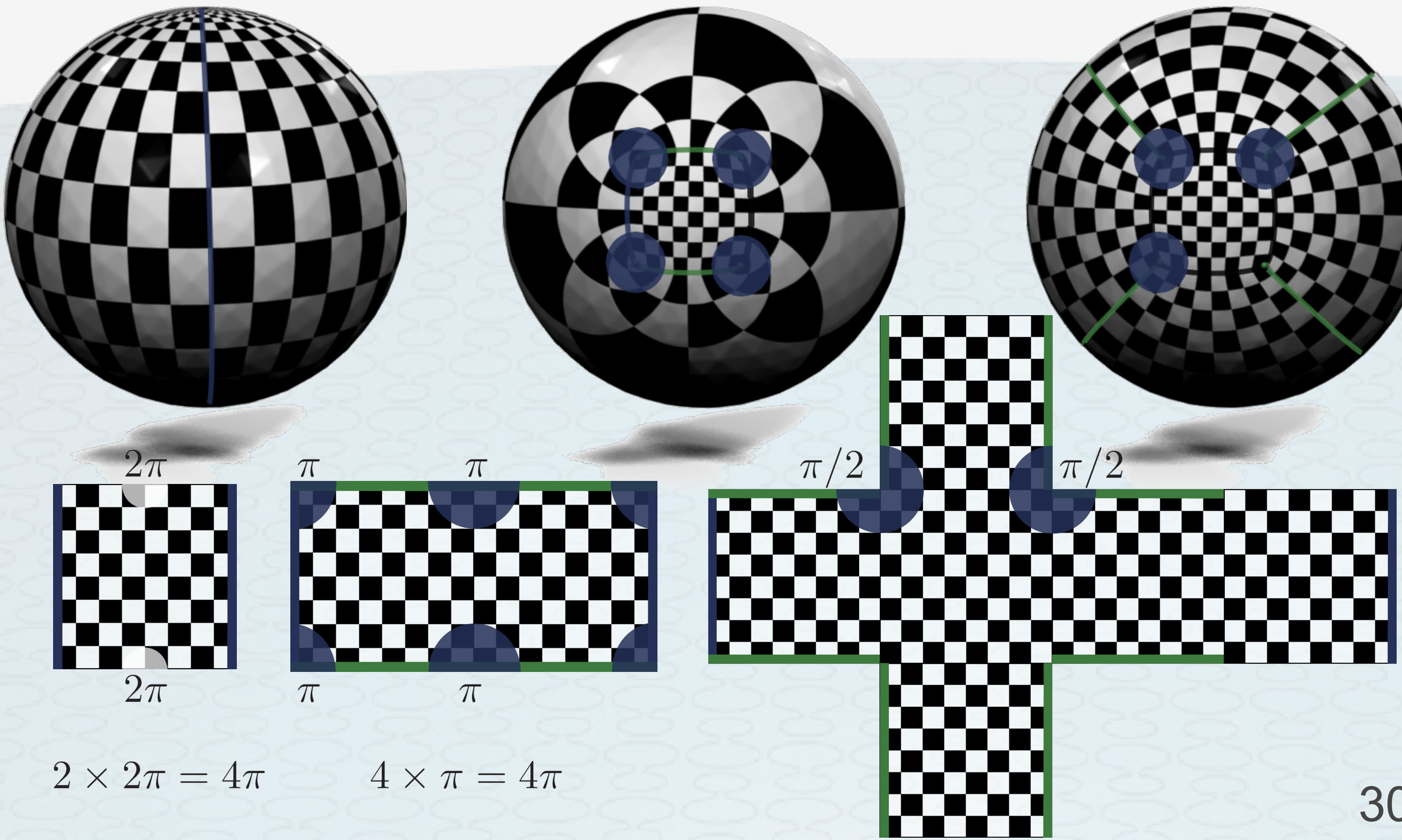
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



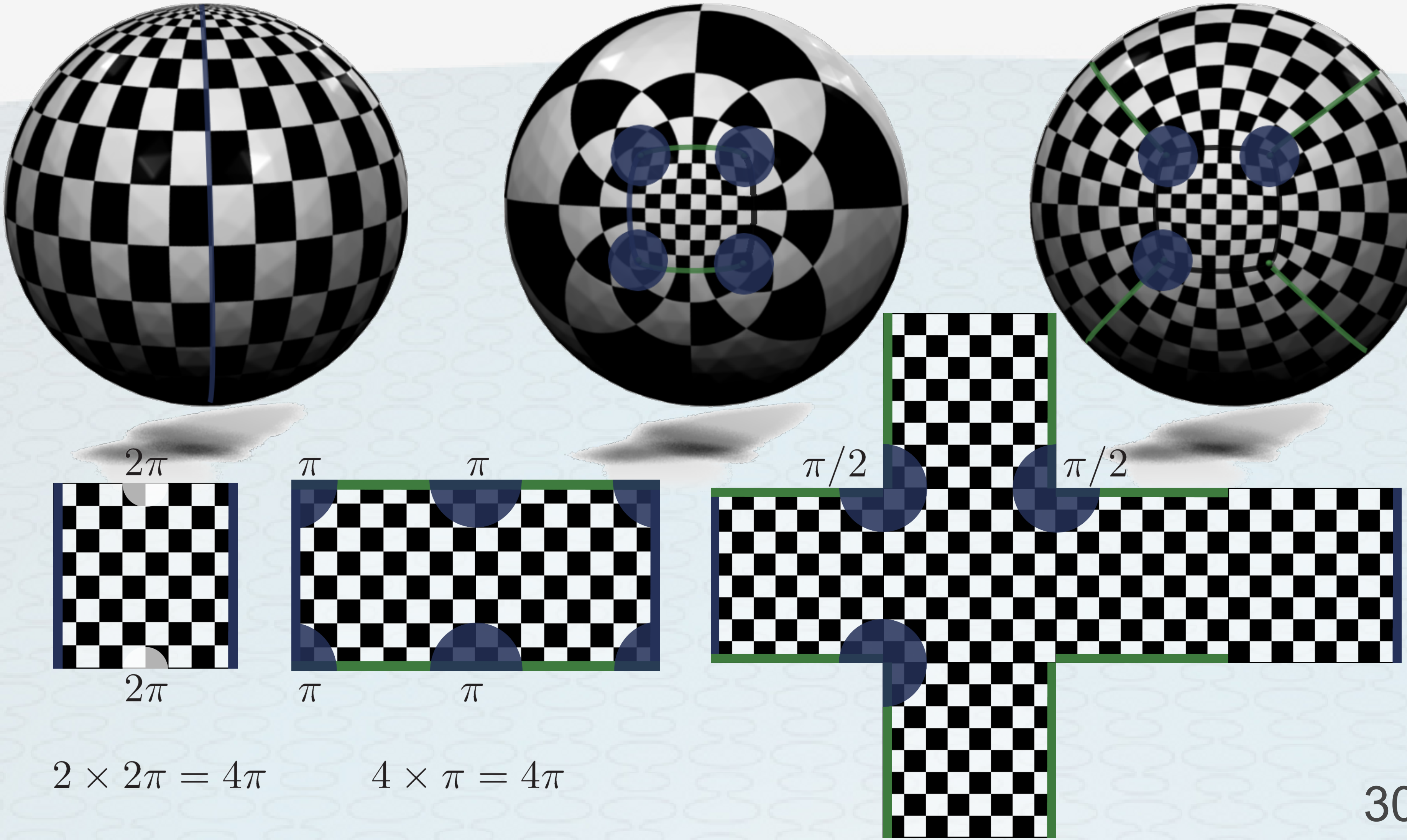
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?

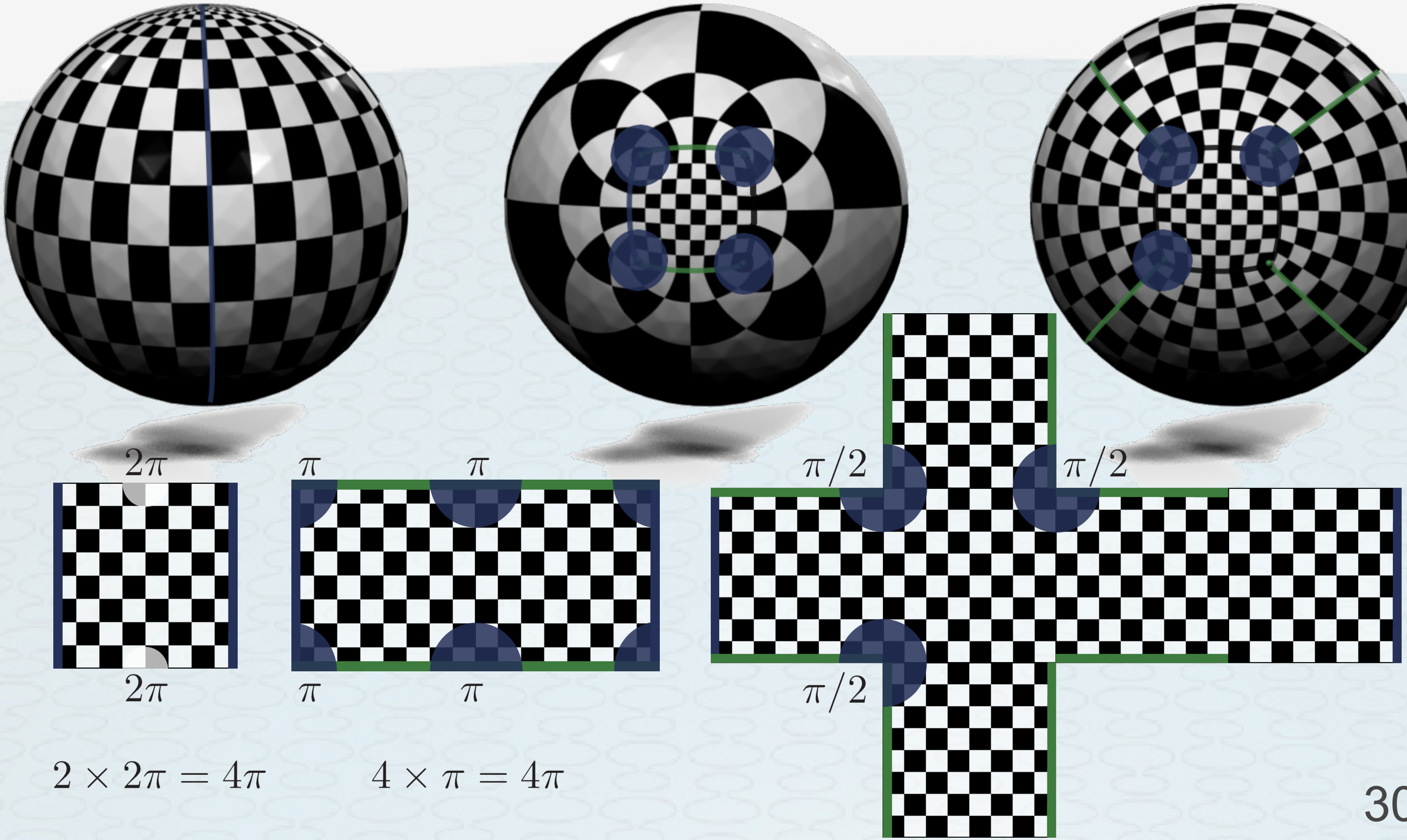


$2 \times 2\pi = 4\pi$

$4 \times \pi = 4\pi$

Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?

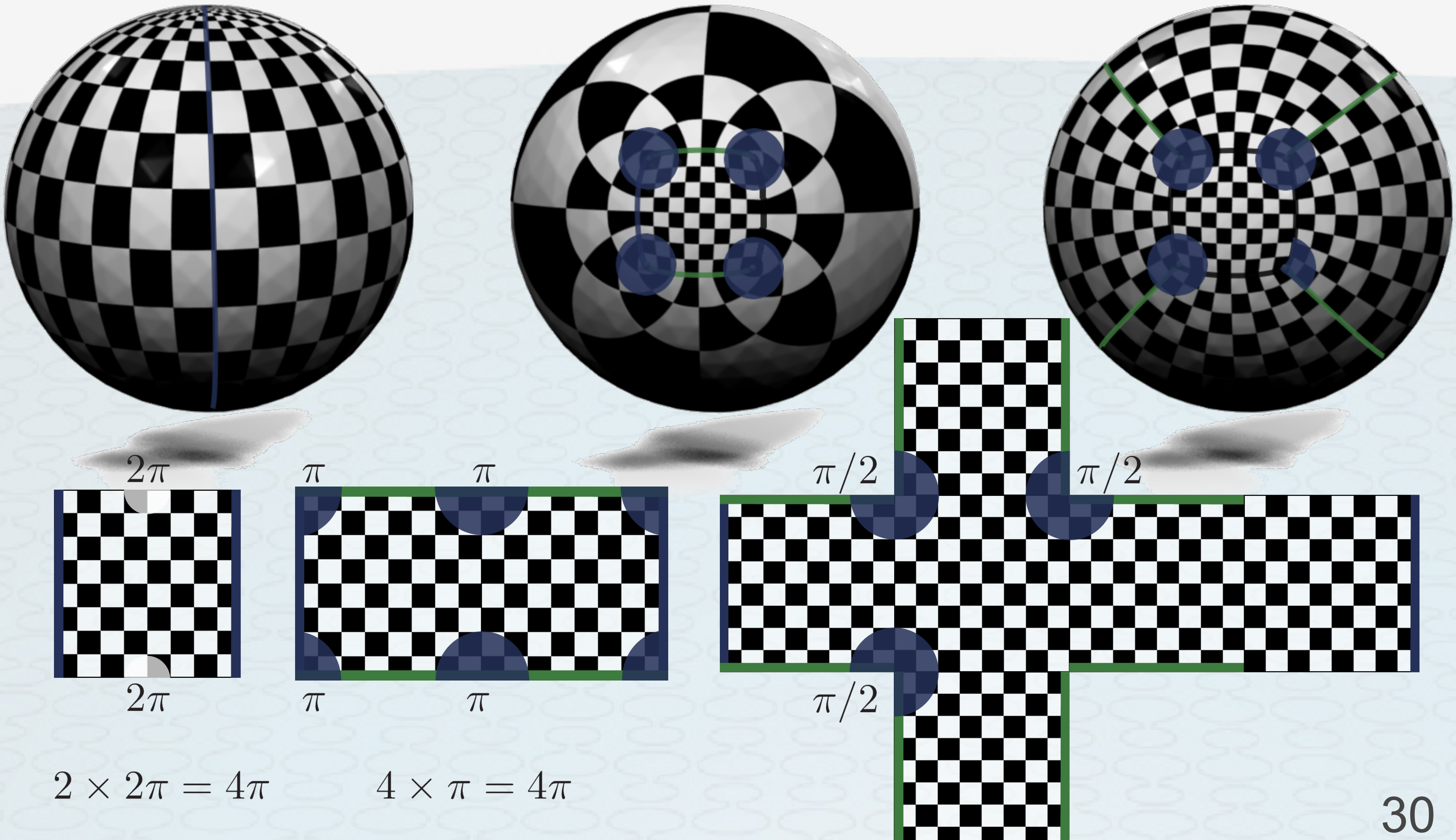


$2 \times 2\pi = 4\pi$

$4 \times \pi = 4\pi$

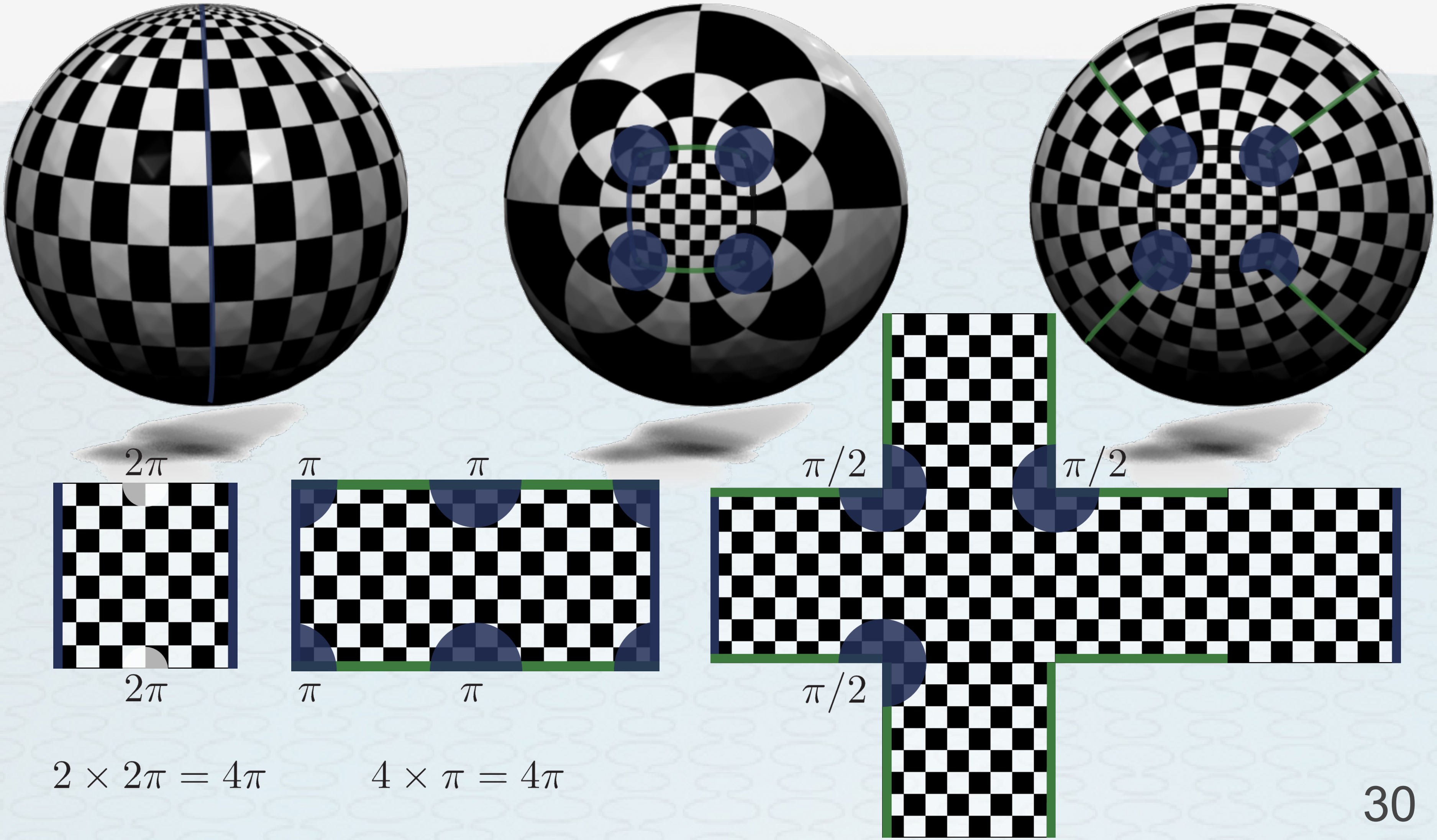
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



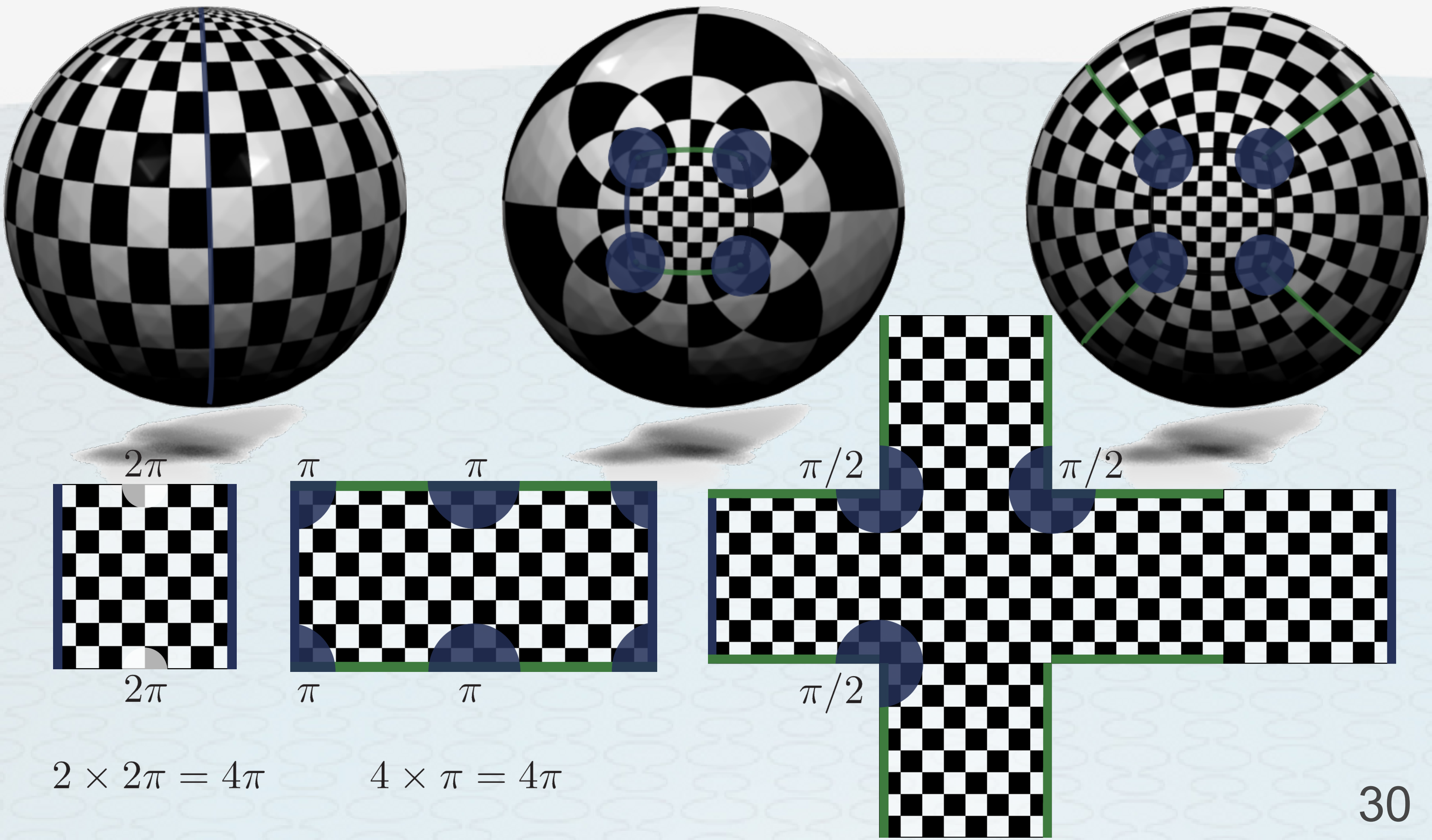
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?

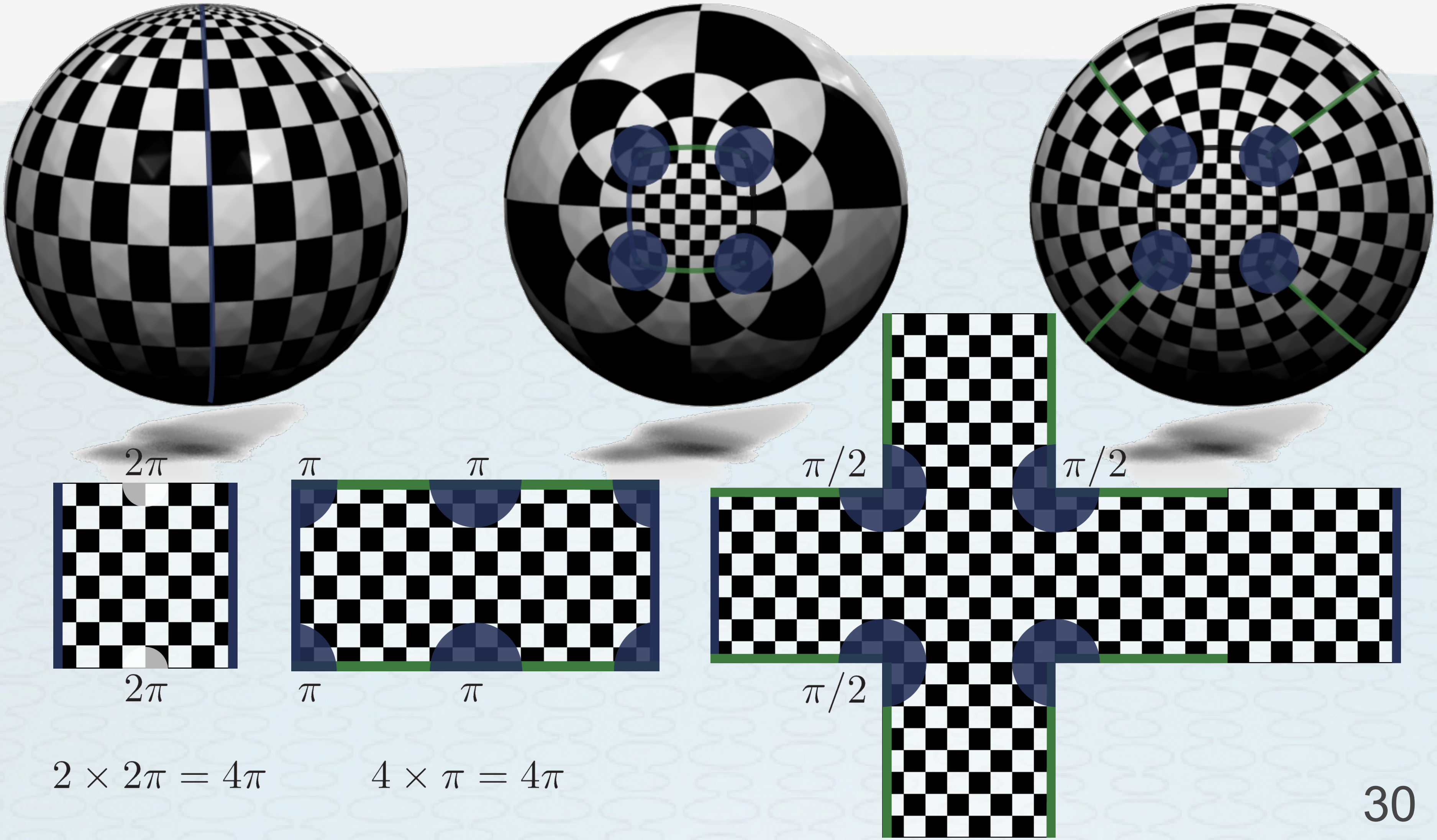


$2 \times 2\pi = 4\pi$

$4 \times \pi = 4\pi$

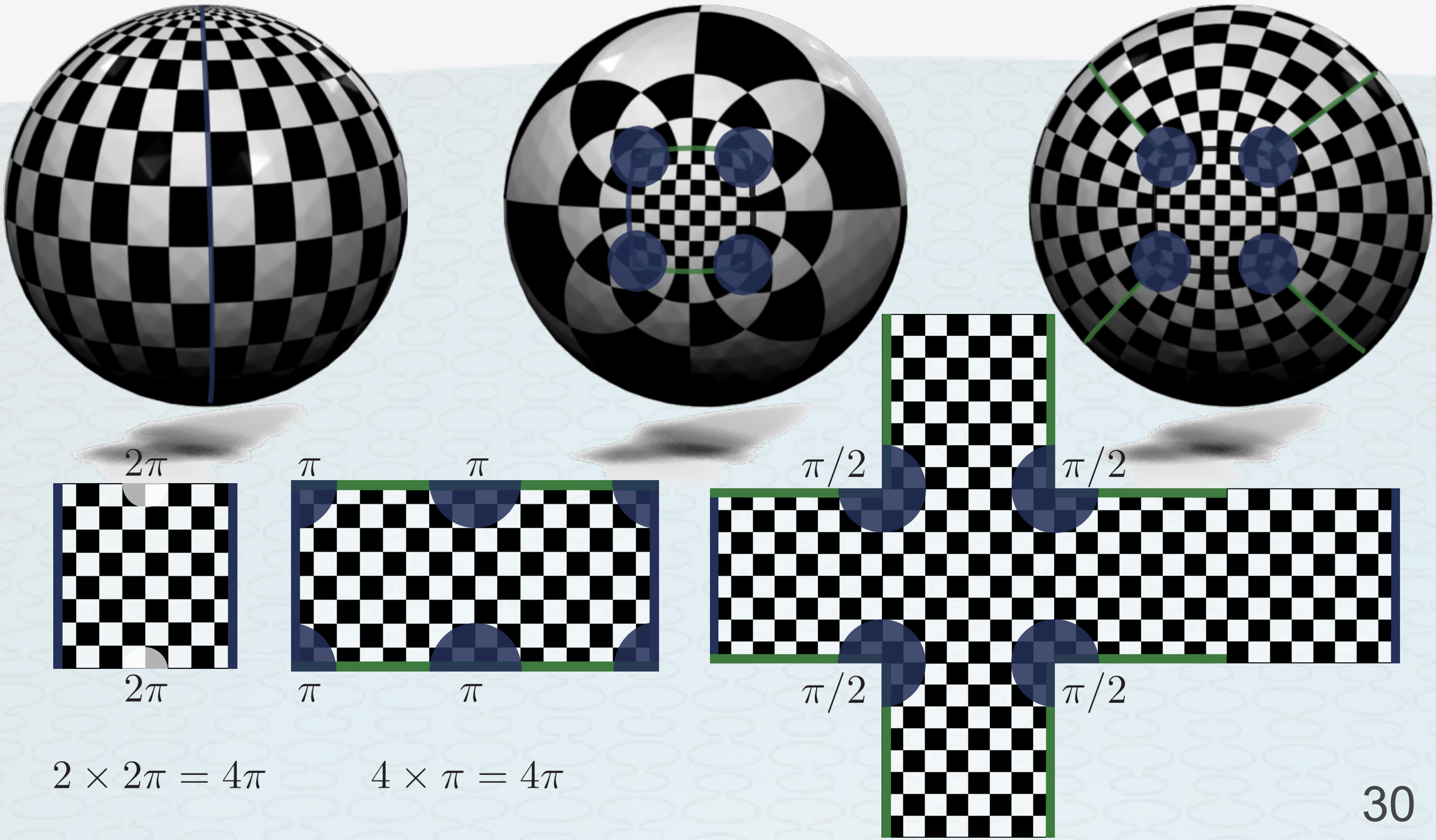
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



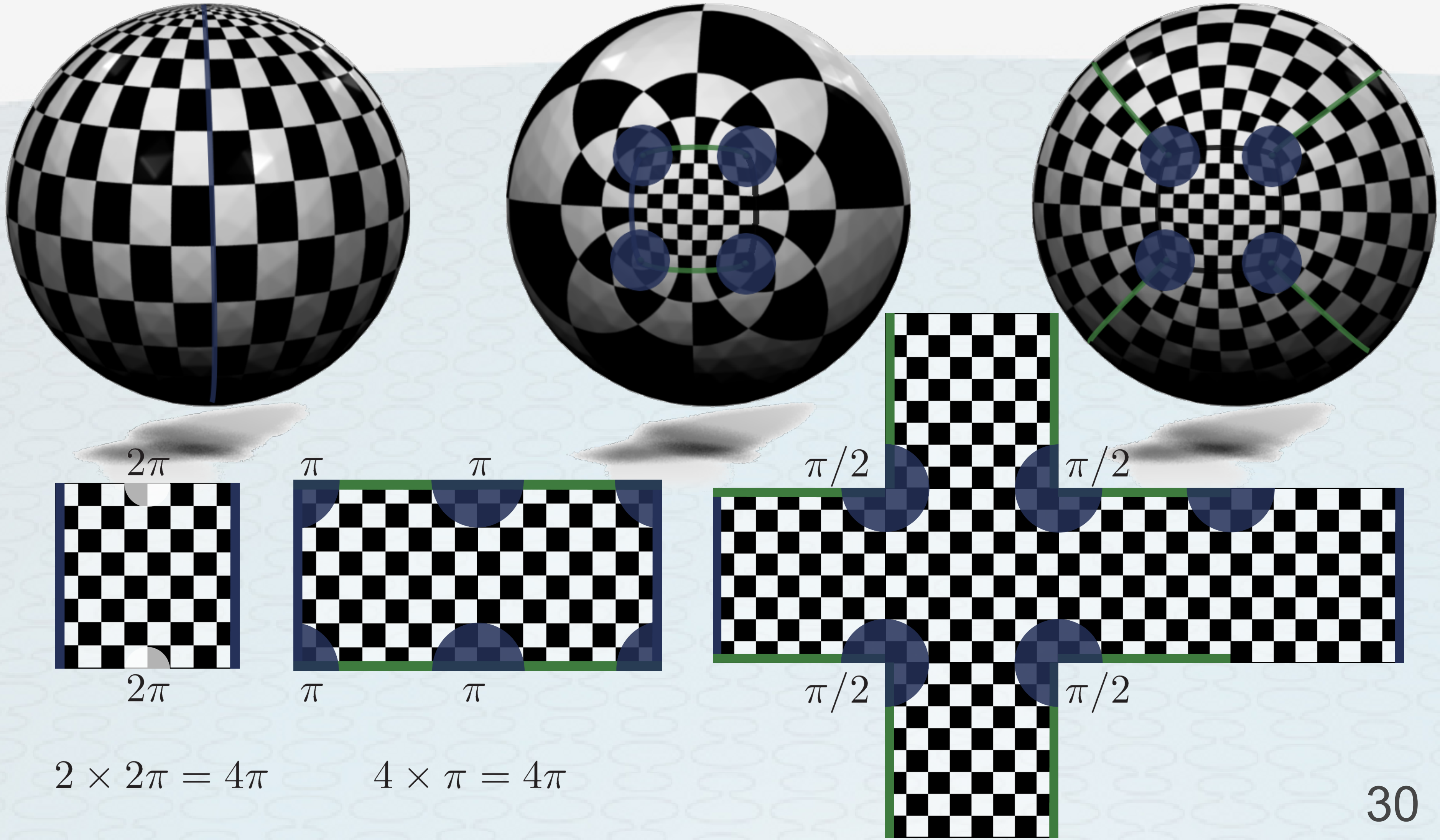
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



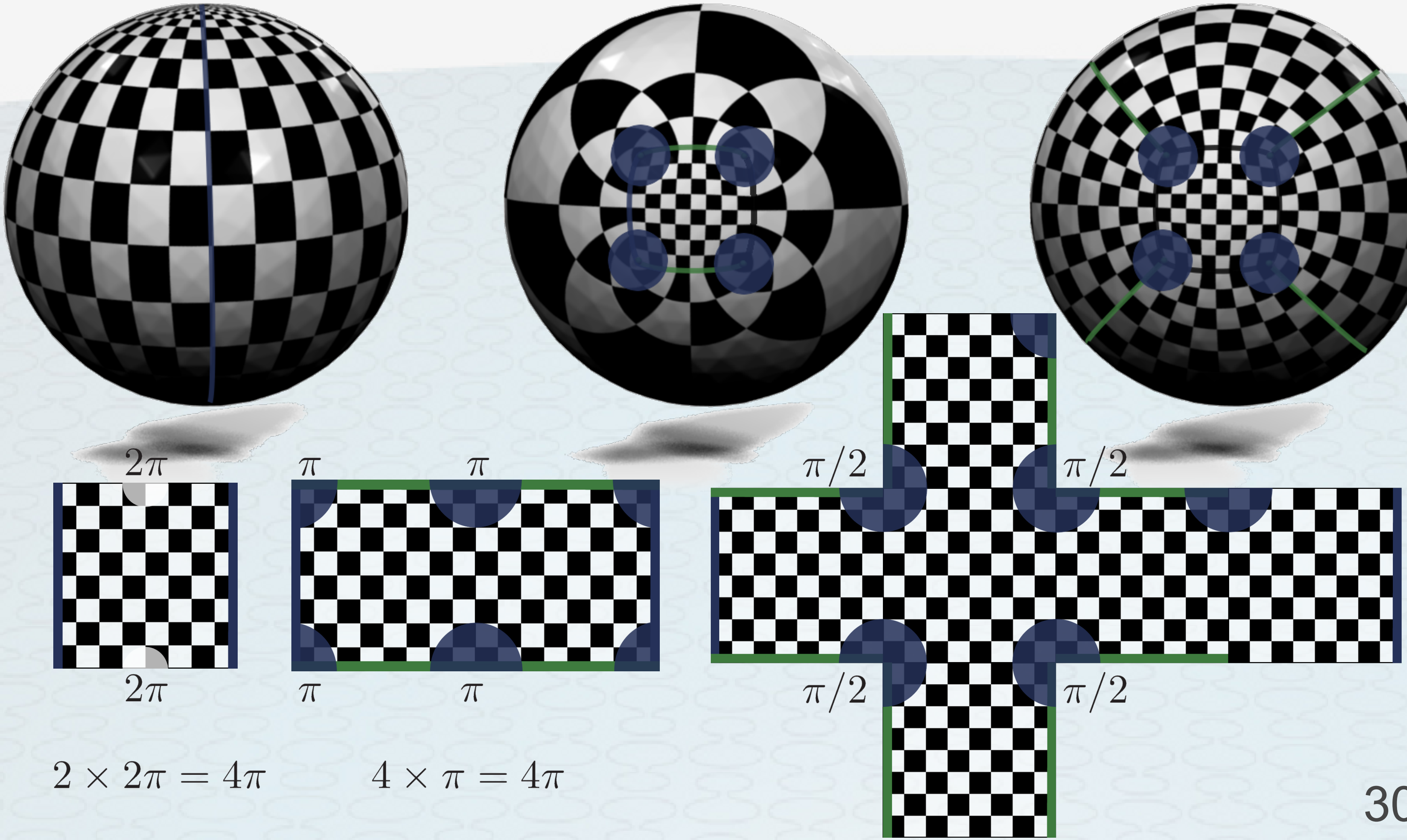
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?

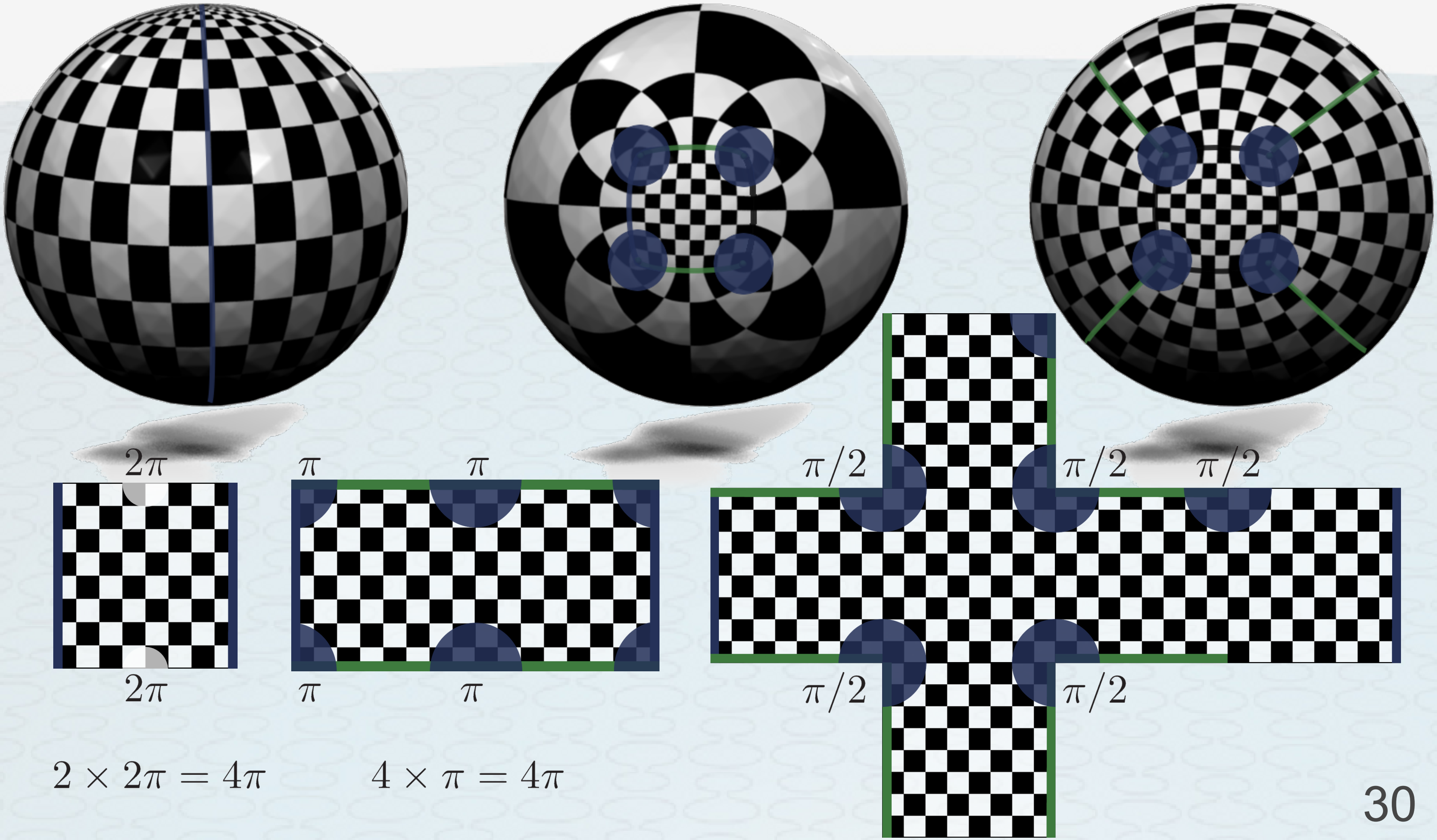


$2 \times 2\pi = 4\pi$

$4 \times \pi = 4\pi$

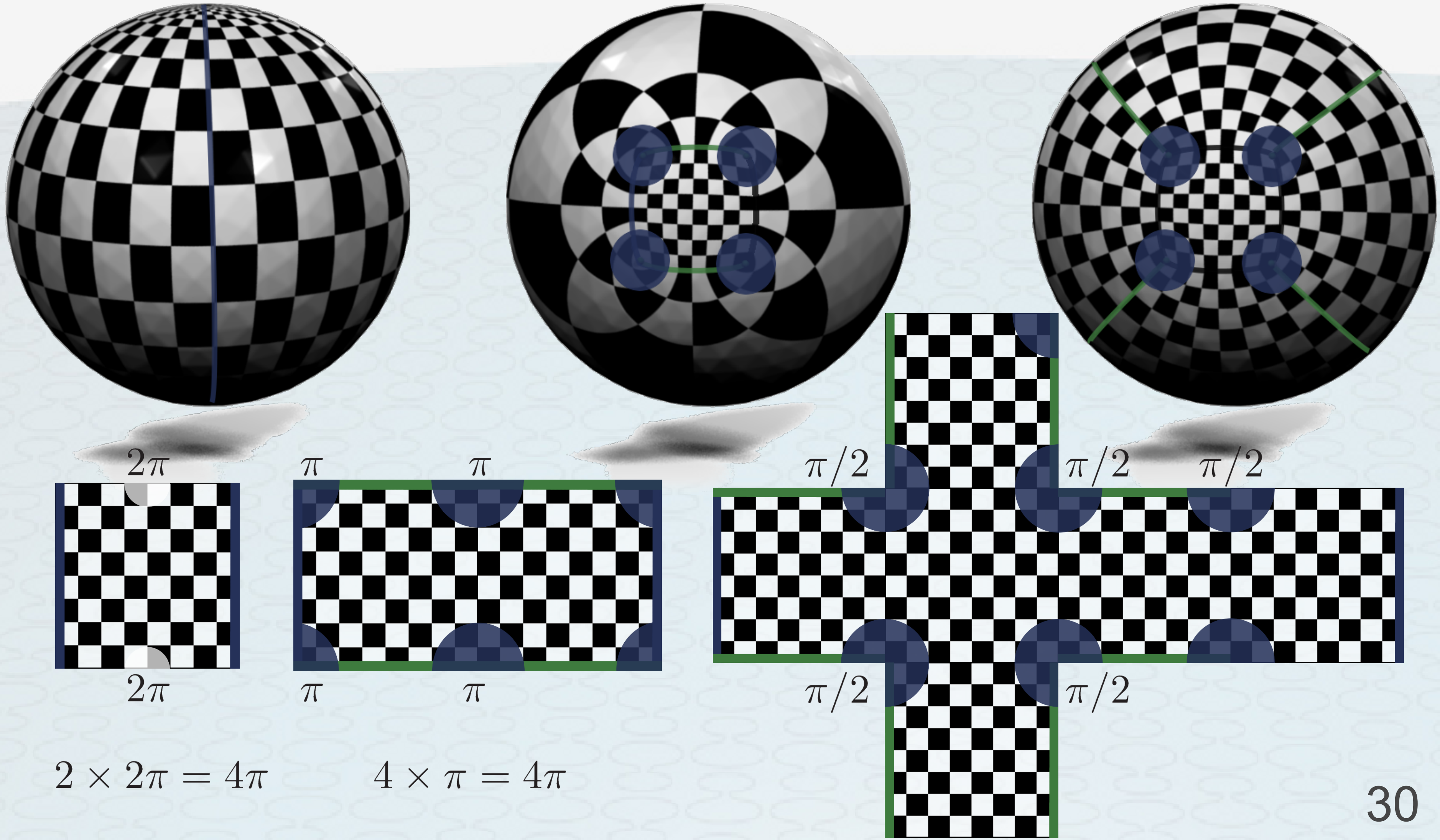
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



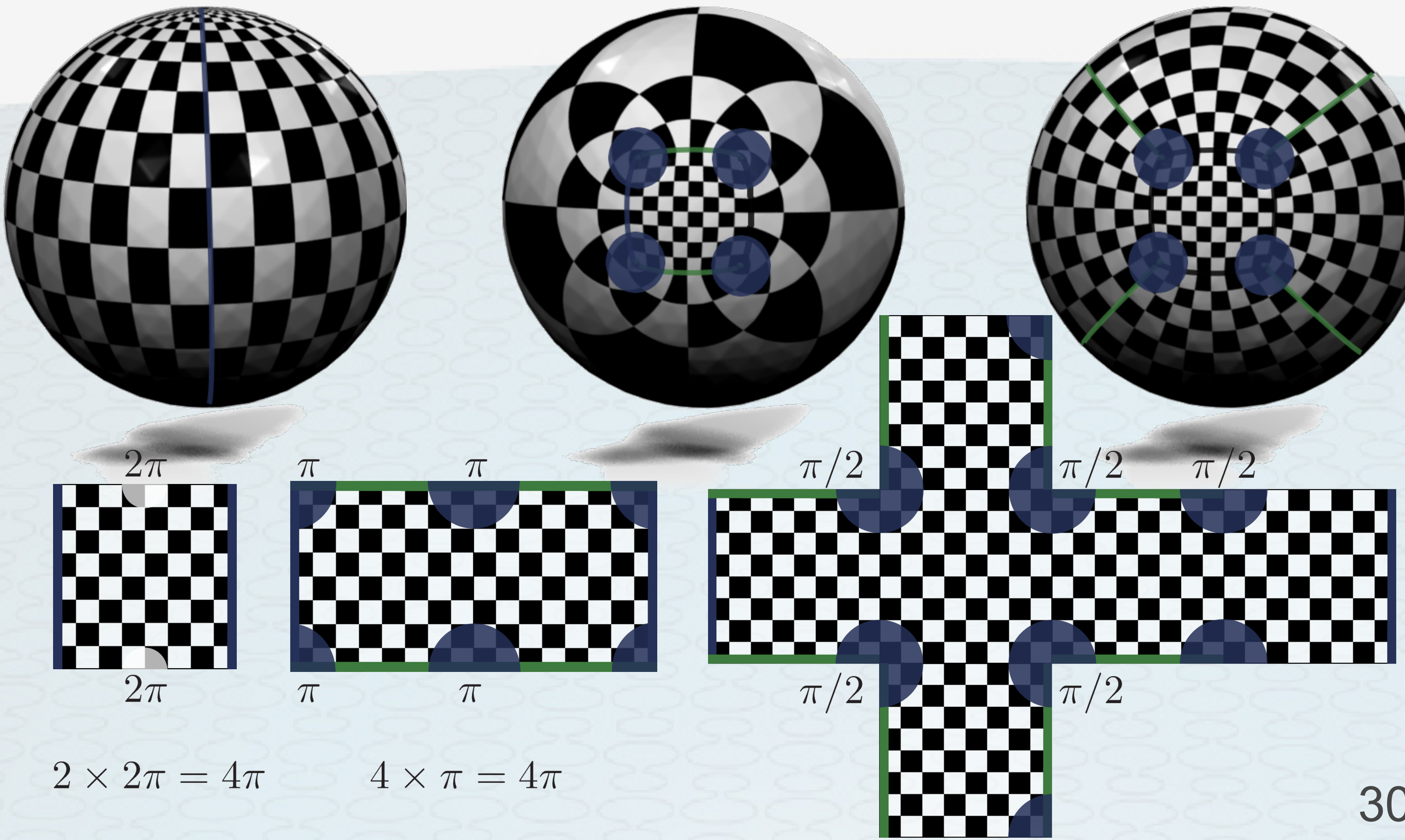
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



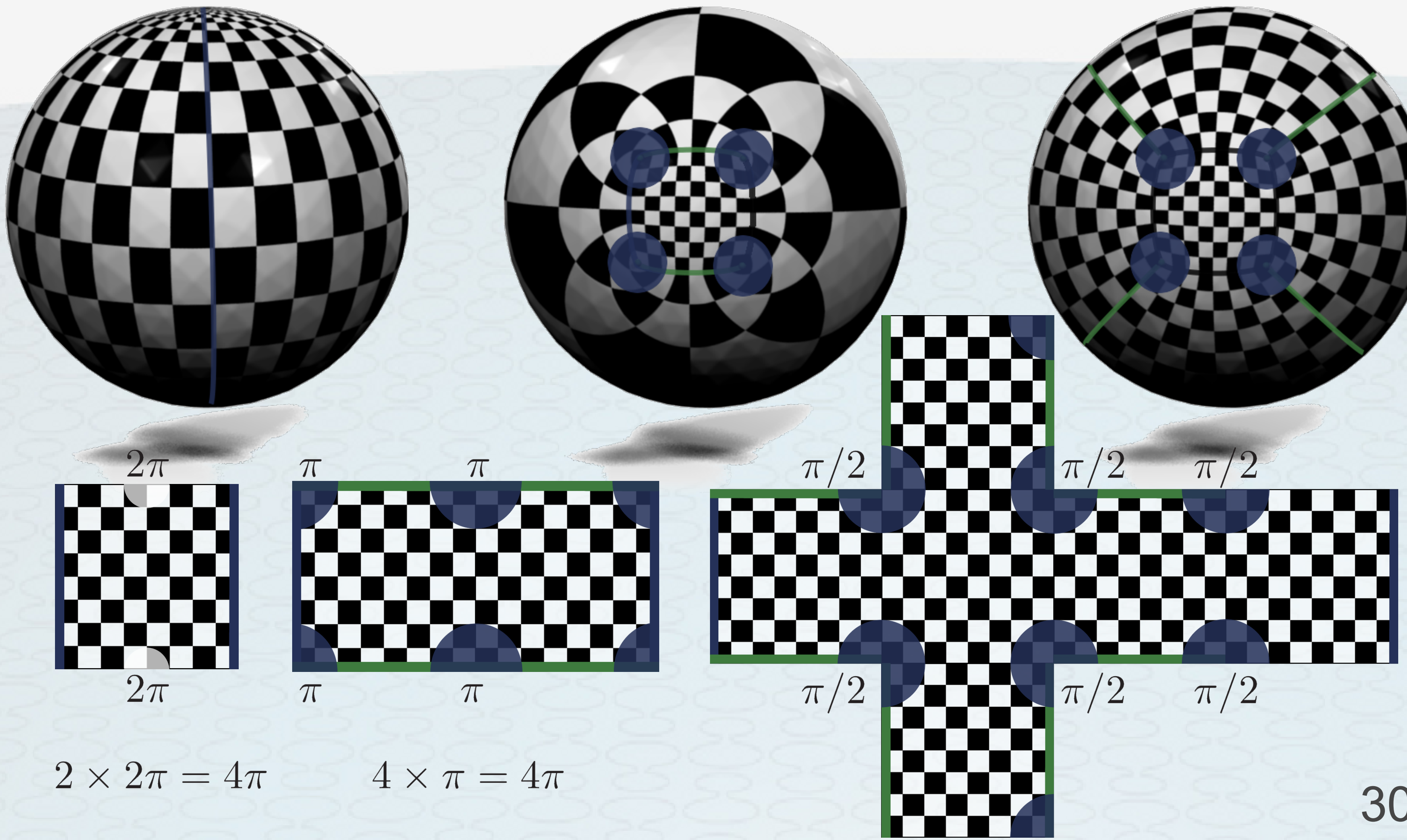
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



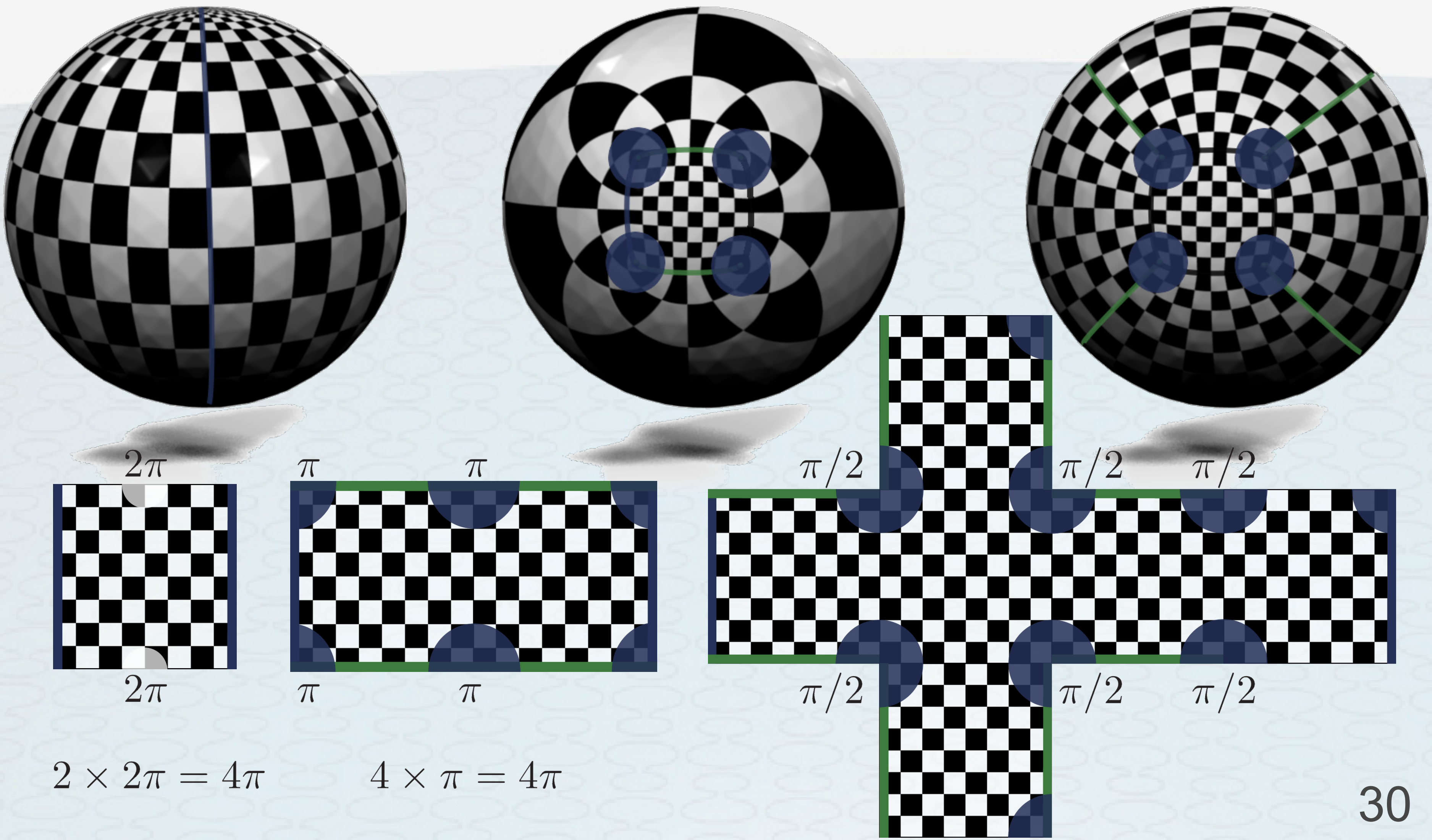
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



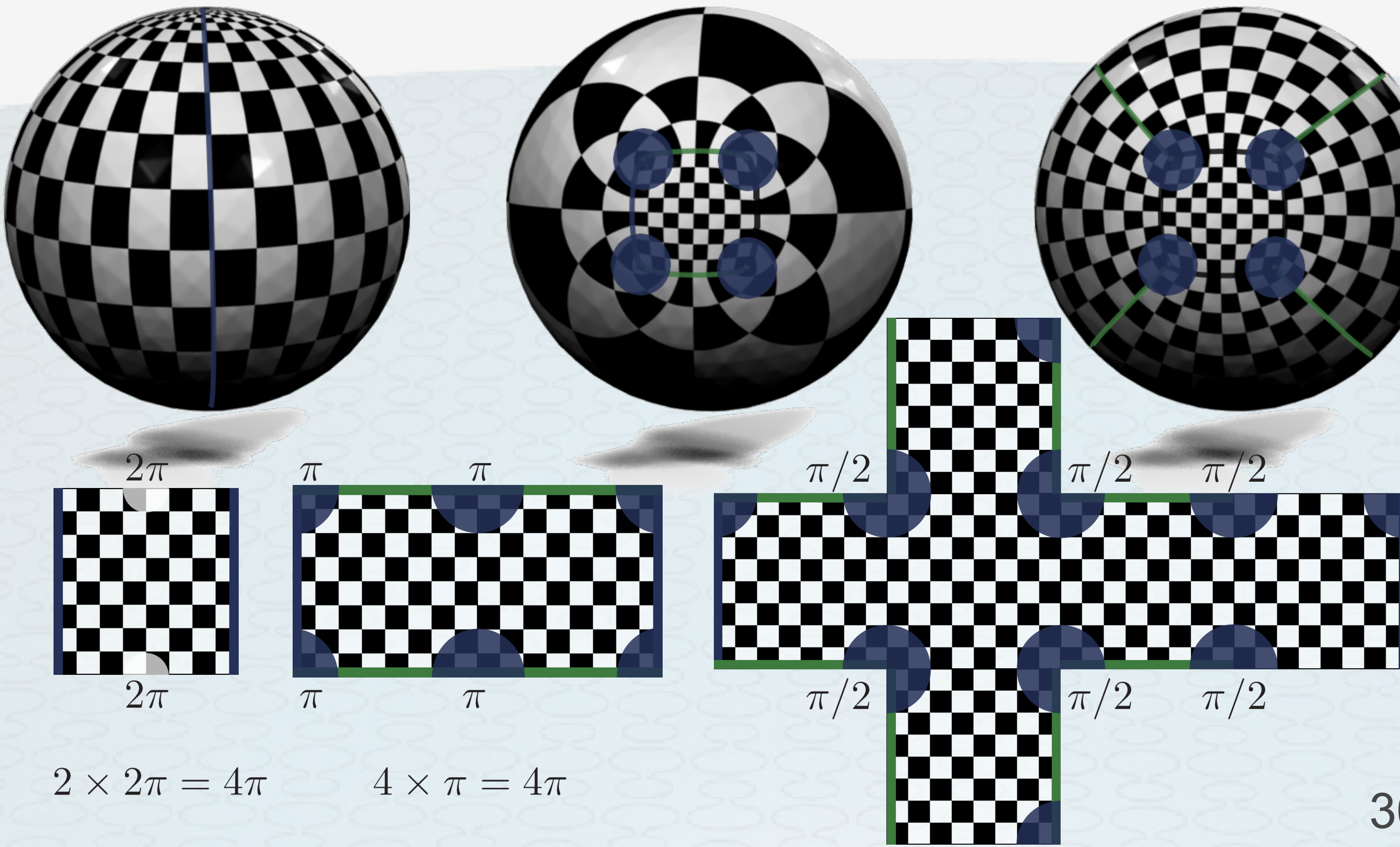
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



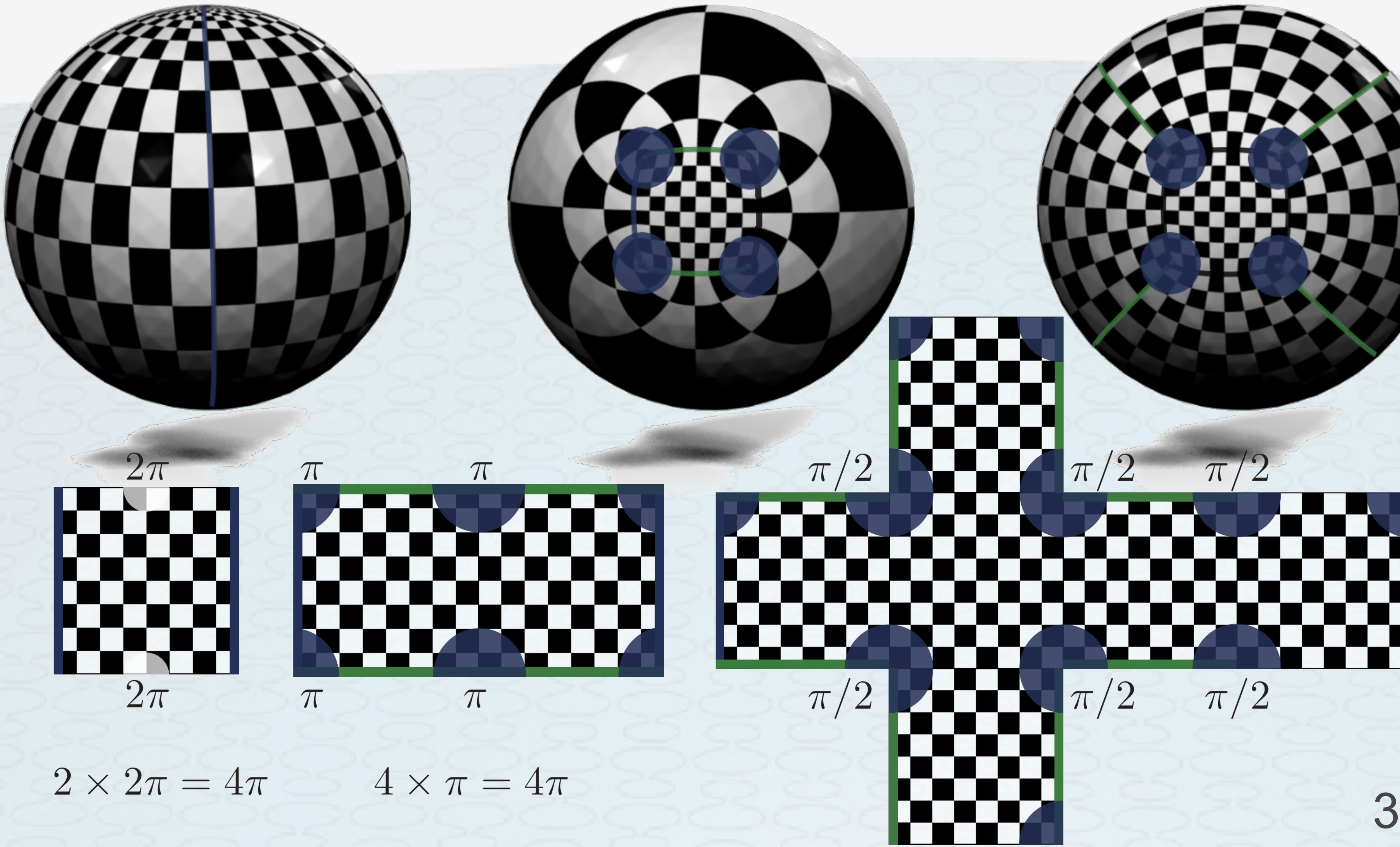
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



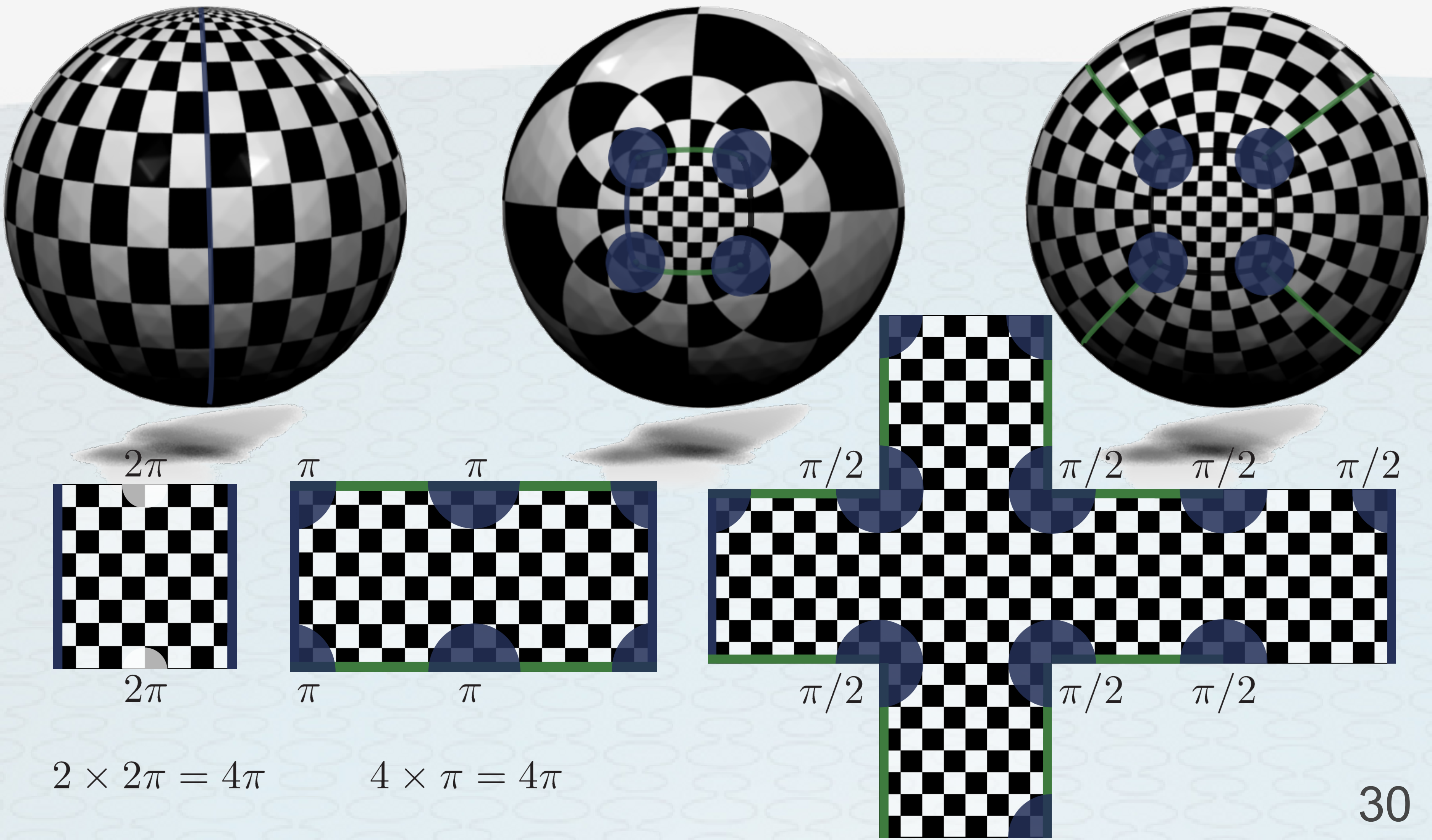
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



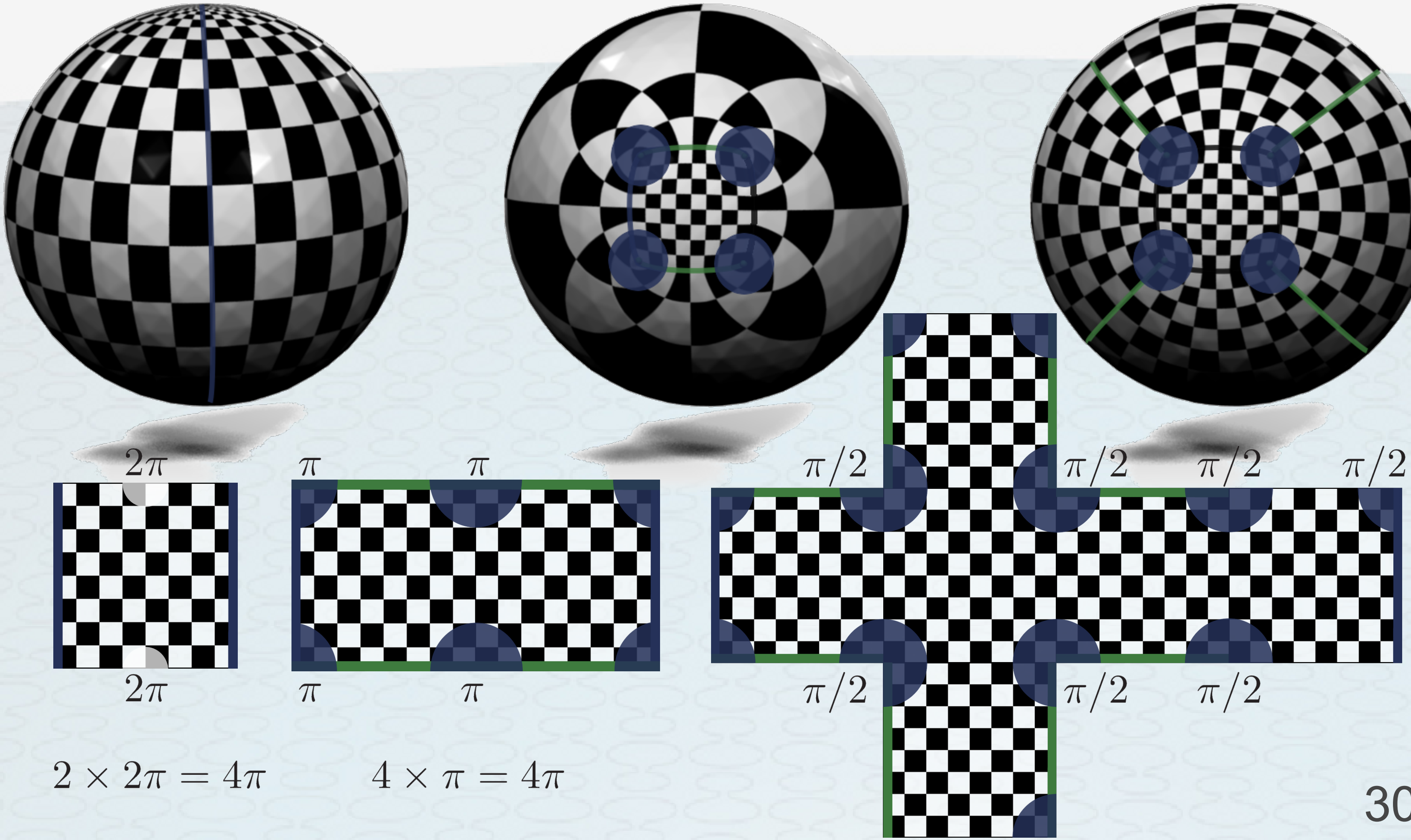
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



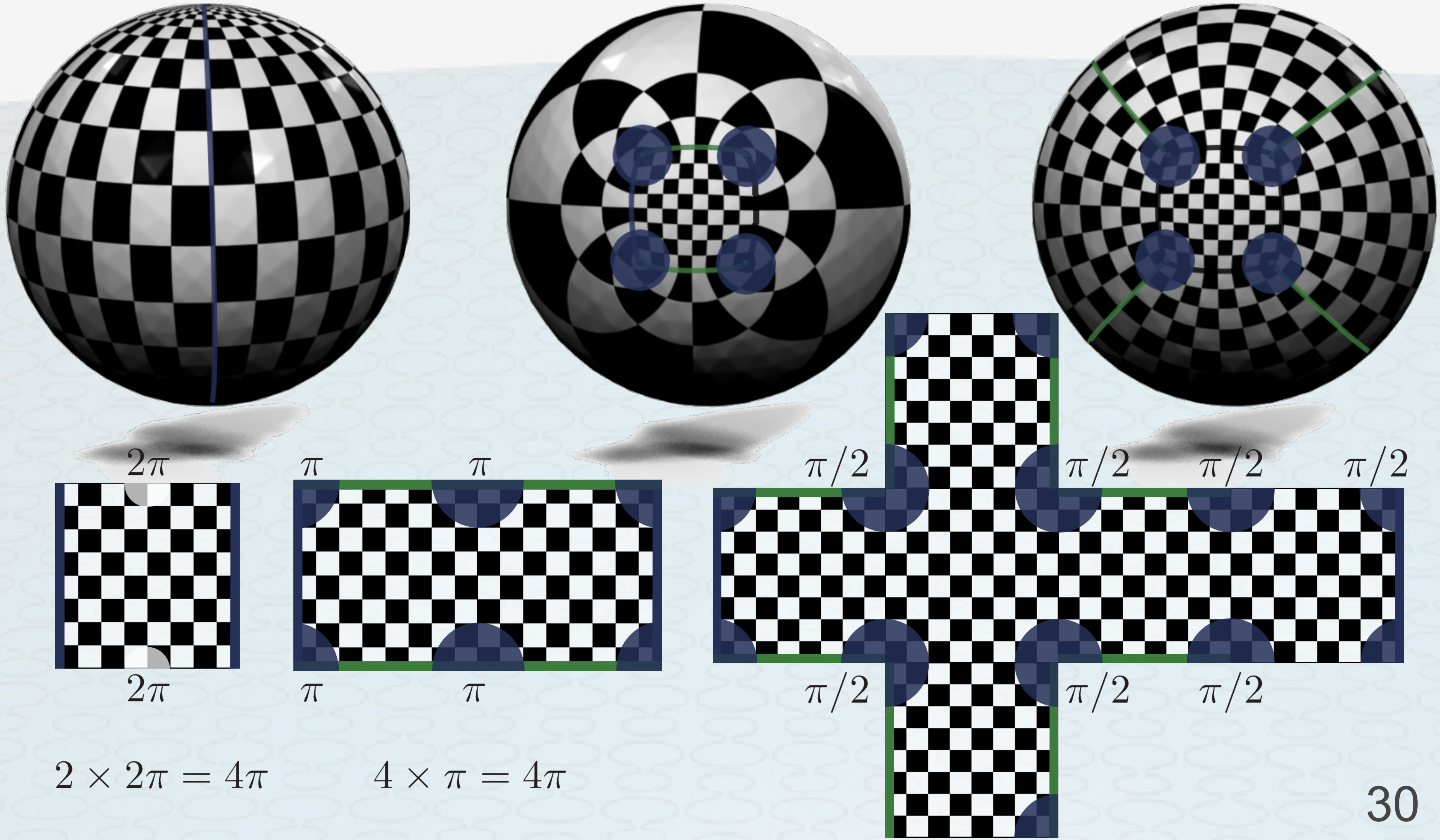
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



Gauss Bonnet theorem revisited (1/2)

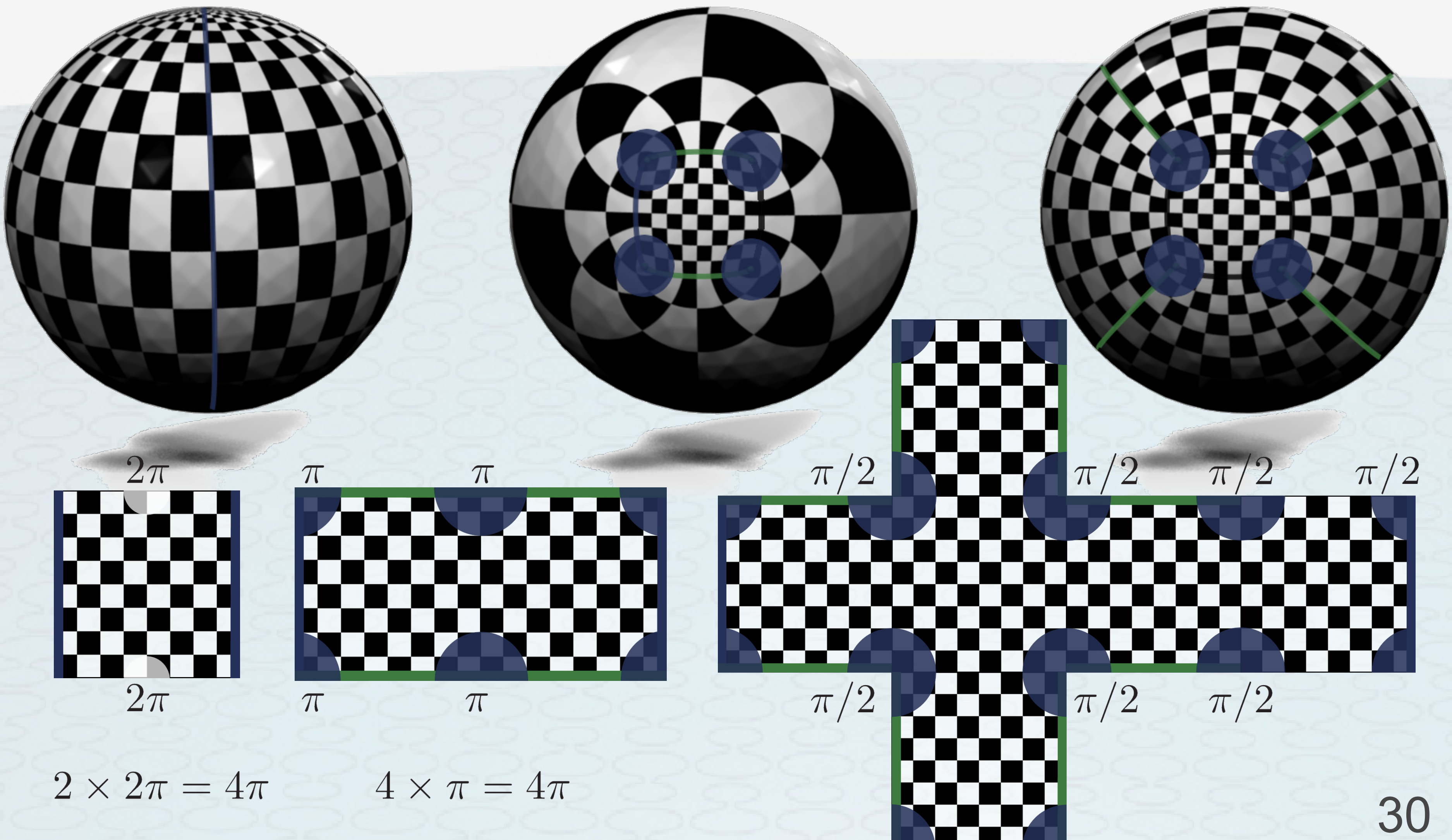
- How many *elastic* quadrilaterals do I need to wrap a sphere?



$2 \times 2\pi = 4\pi$ $\quad$ $4 \times \pi = 4\pi$

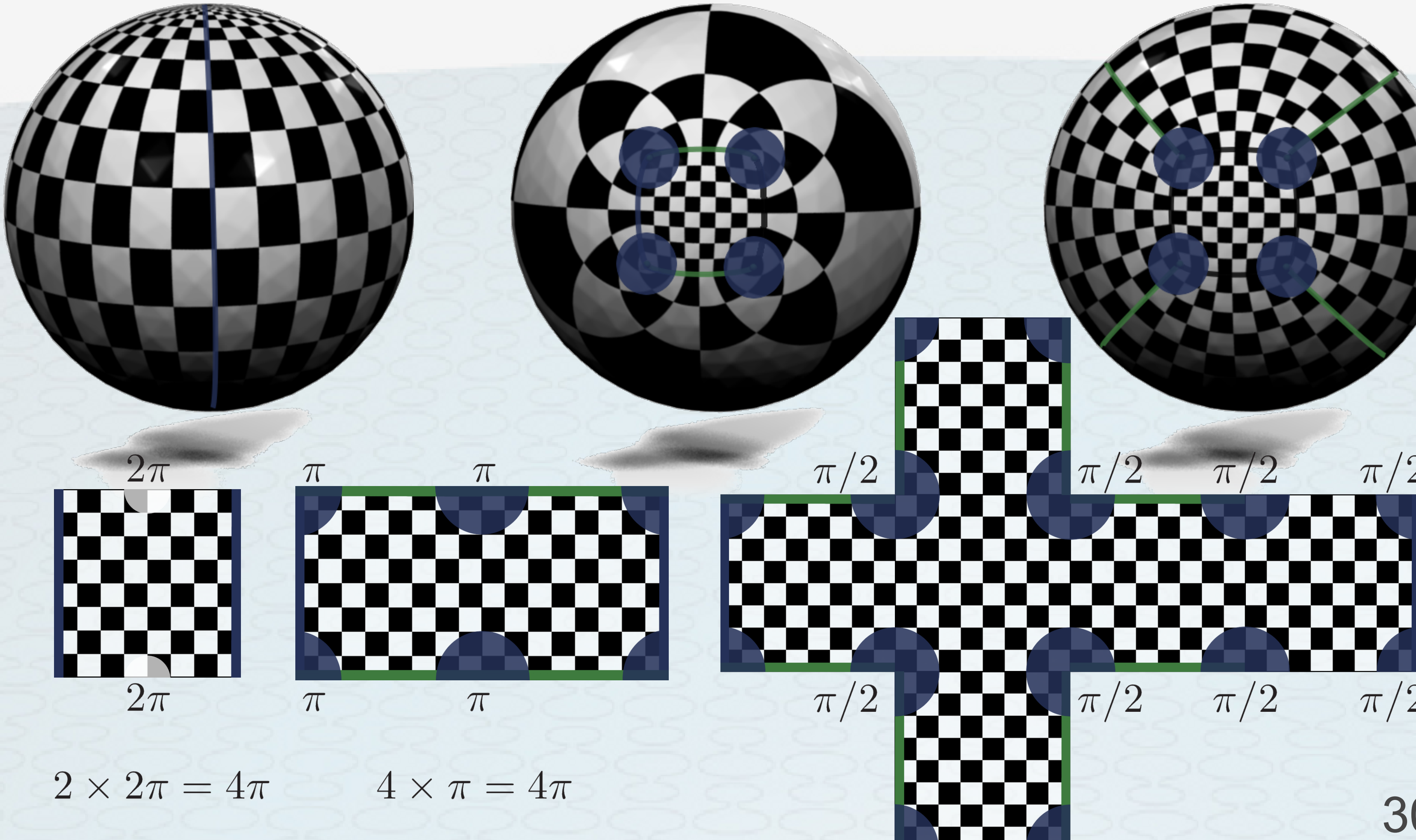
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



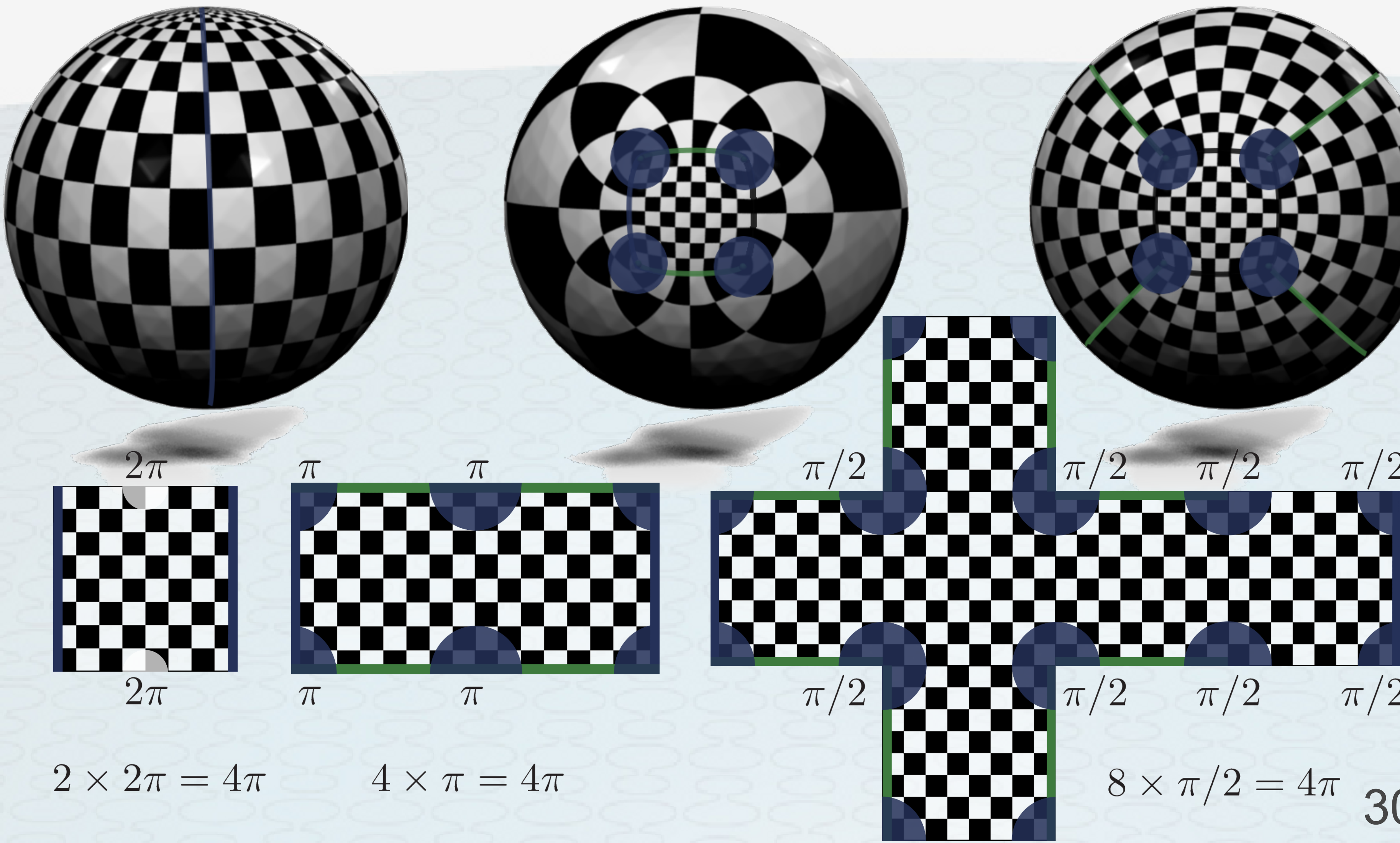
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



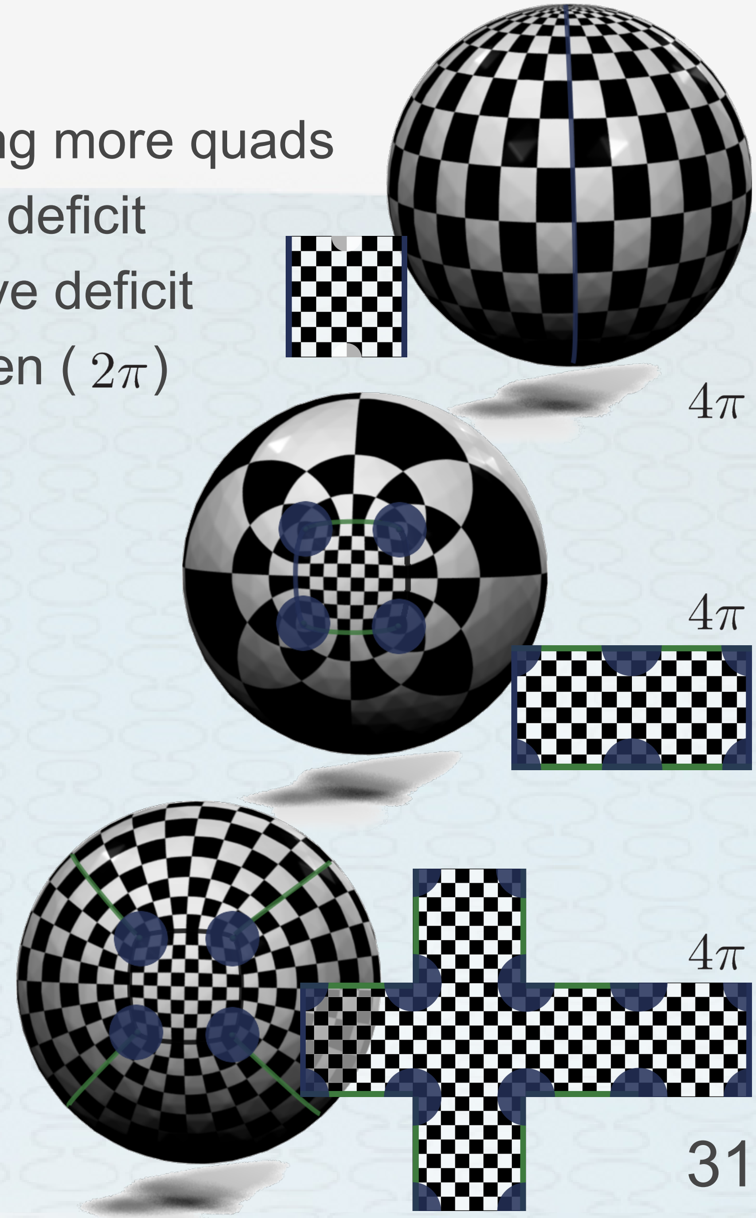
Gauss Bonnet theorem revisited (1/2)

- How many *elastic* quadrilaterals do I need to wrap a sphere?



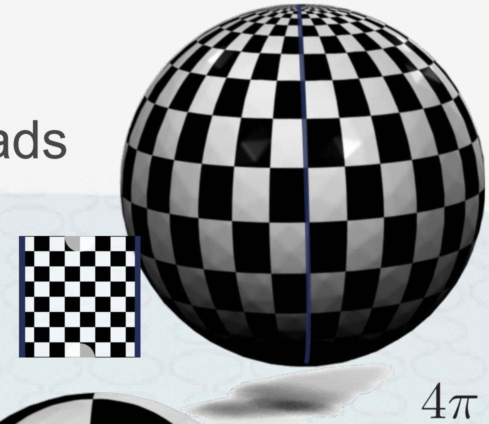
Gauss Bonnet theorem revisited (2/2)

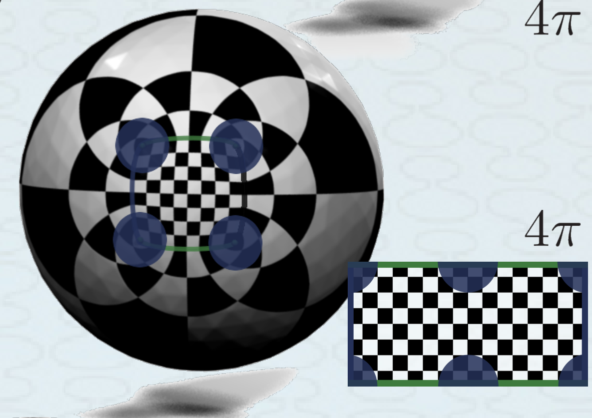
- This phenomenon continues when adding more quads
  - Corner of less than 4 quads: positive deficit
  - Corner of more than 4 quads: negative deficit
  - Same reasoning if the atlas is cut open (2π)
- What's going on?

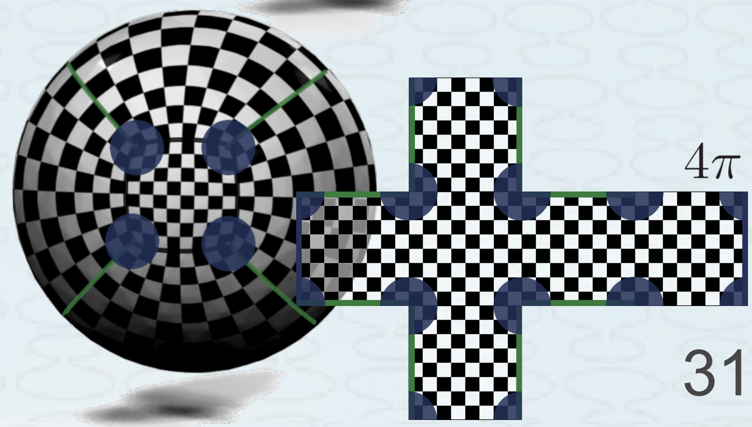


Gauss Bonnet theorem revisited (2/2)

- This phenomenon continues when adding more quads
  - Corner of less than 4 quads: positive deficit
  - Corner of more than 4 quads: negative deficit
  - Same reasoning if the atlas is cut open (2π)
- What's going on?
  - Gauss Bonnet theorem, for closed surfaces
    - $\int_{\mathcal{S}} K dA = 2\pi\chi(\mathcal{S})$
  - **Constraints for optimization problems**

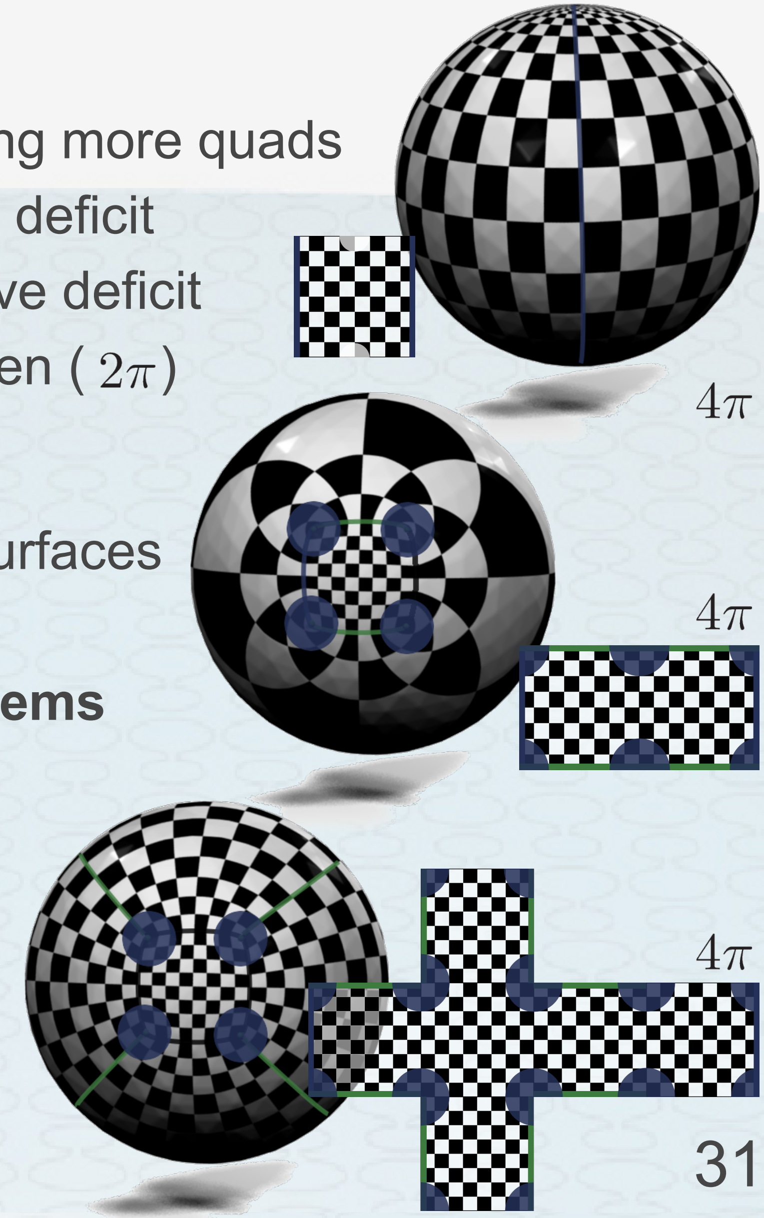






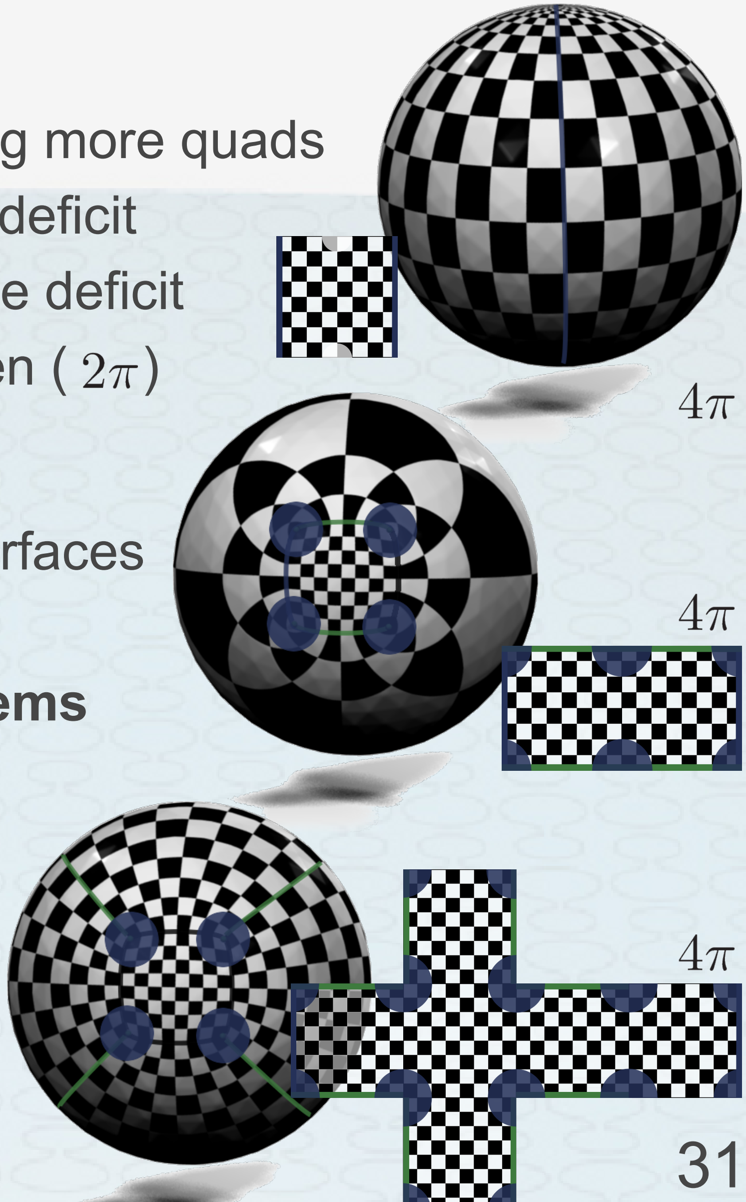
Gauss Bonnet theorem revisited (2/2)

- This phenomenon continues when adding more quads
  - Corner of less than 4 quads: positive deficit
  - Corner of more than 4 quads: negative deficit
  - Same reasoning if the atlas is cut open (2π)
- What's going on?
  - Gauss Bonnet theorem, for closed surfaces
    - $\int_{\mathcal{S}} K dA = 2\pi\chi(\mathcal{S})$
  - **Constraints for optimization problems**
- Btw, how did the others do?

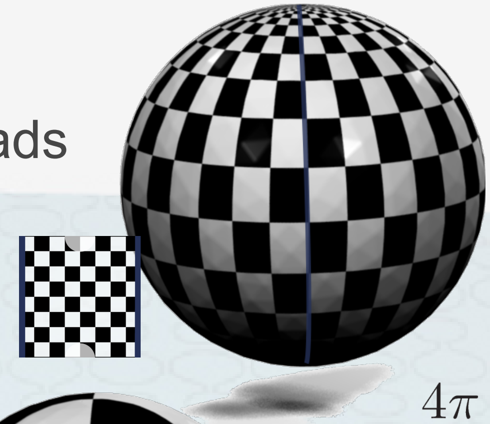


Gauss Bonnet theorem revisited (2/2)

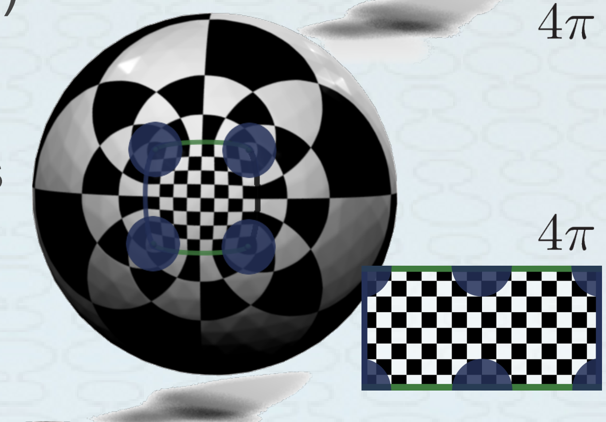
- This phenomenon continues when adding more quads
  - Corner of less than 4 quads: positive deficit
  - Corner of more than 4 quads: negative deficit
  - Same reasoning if the atlas is cut open (2π)
- What's going on?
  - Gauss Bonnet theorem, for closed surfaces
    - $\int_{\mathcal{S}} K dA = 2\pi\chi(\mathcal{S})$
  - **Constraints for optimization problems**
- Btw, how did the others do?
  - Vector fields, Poincare-Hopf theorem
    - $\sum_i index(x_i) = \chi(\mathcal{S})$



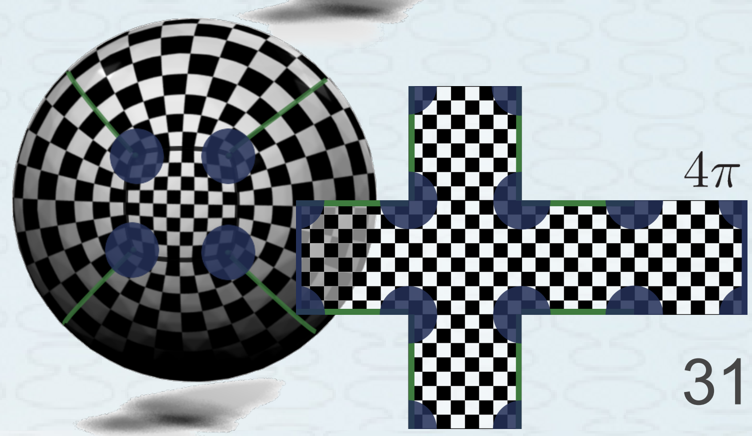
Gauss Bonnet theorem revisited (2/2)



- This phenomenon continues when adding more quads
  - Corner of less than 4 quads: positive deficit
  - Corner of more than 4 quads: negative deficit
  - Same reasoning if the atlas is cut open (2π)

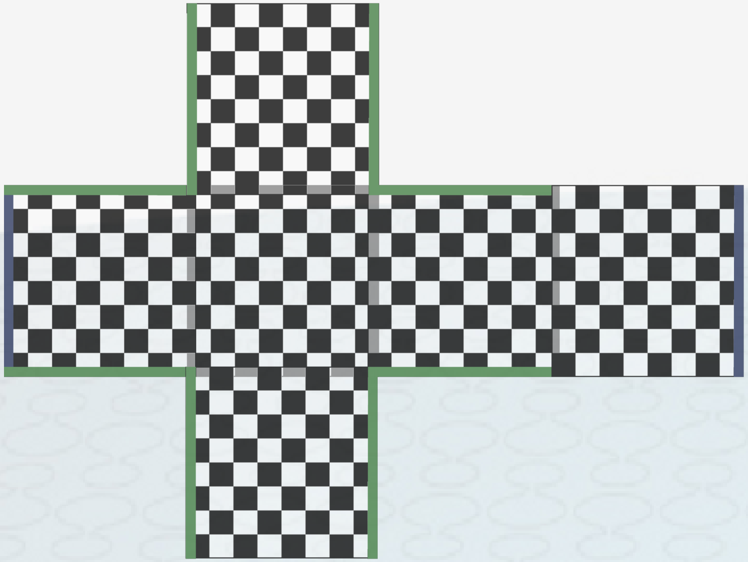


- What's going on?
  - Gauss Bonnet theorem, for closed surfaces
    - $\int_{\mathcal{S}} K dA = 2\pi\chi(\mathcal{S})$
  - **Constraints for optimization problems**



- Btw, how did the others do?
  - Vector fields, Poincare-Hopf theorem
    - $\sum_i index(x_i) = \chi(\mathcal{S})$
  - Scalar fields, Morse-Euler relation
    - $\sum_i \mu_i(-1)^i = \chi(\mathcal{S})$

Uniformization theorem revisited

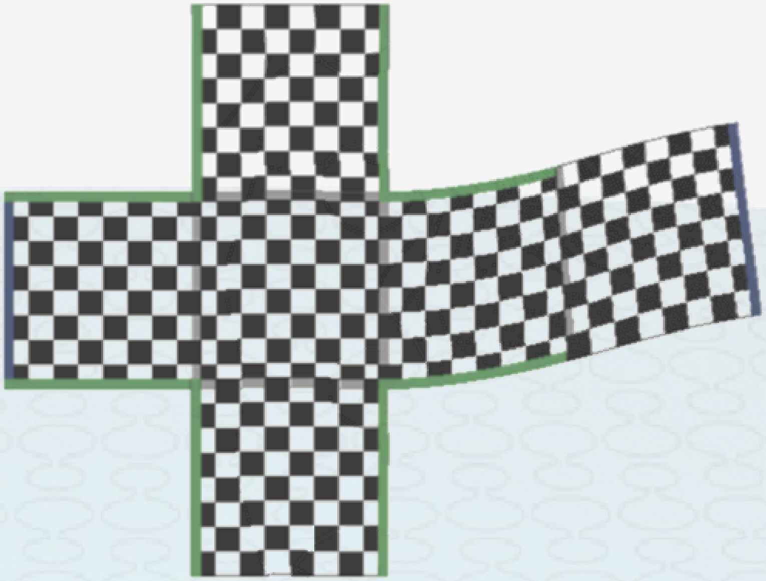


Uniformization theorem revisited

Uniformization theorem revisited

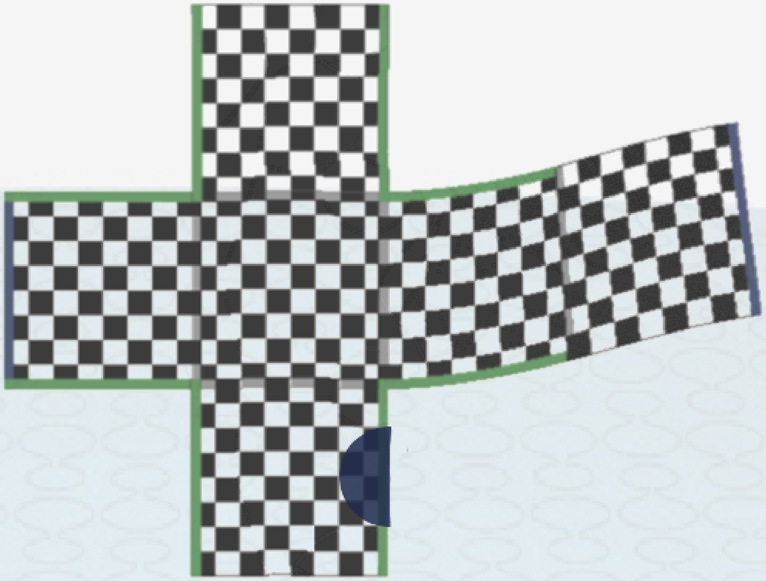
- Except at prescribed singularities, constant Gaussian curvature

Uniformization theorem revisited



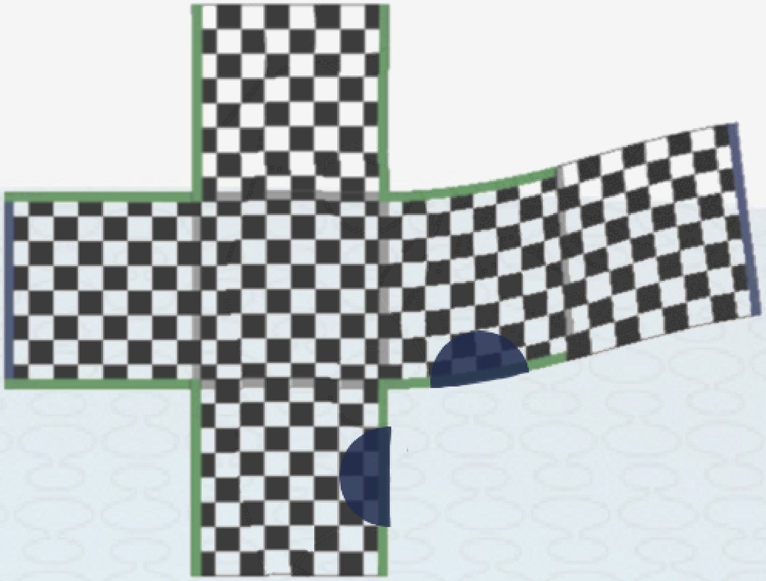
- Except at prescribed singularities, constant Gaussian curvature
- After *a* conformal transformation

Uniformization theorem revisited



- Except at prescribed singularities, constant Gaussian curvature
- After *a* conformal transformation

Uniformization theorem revisited

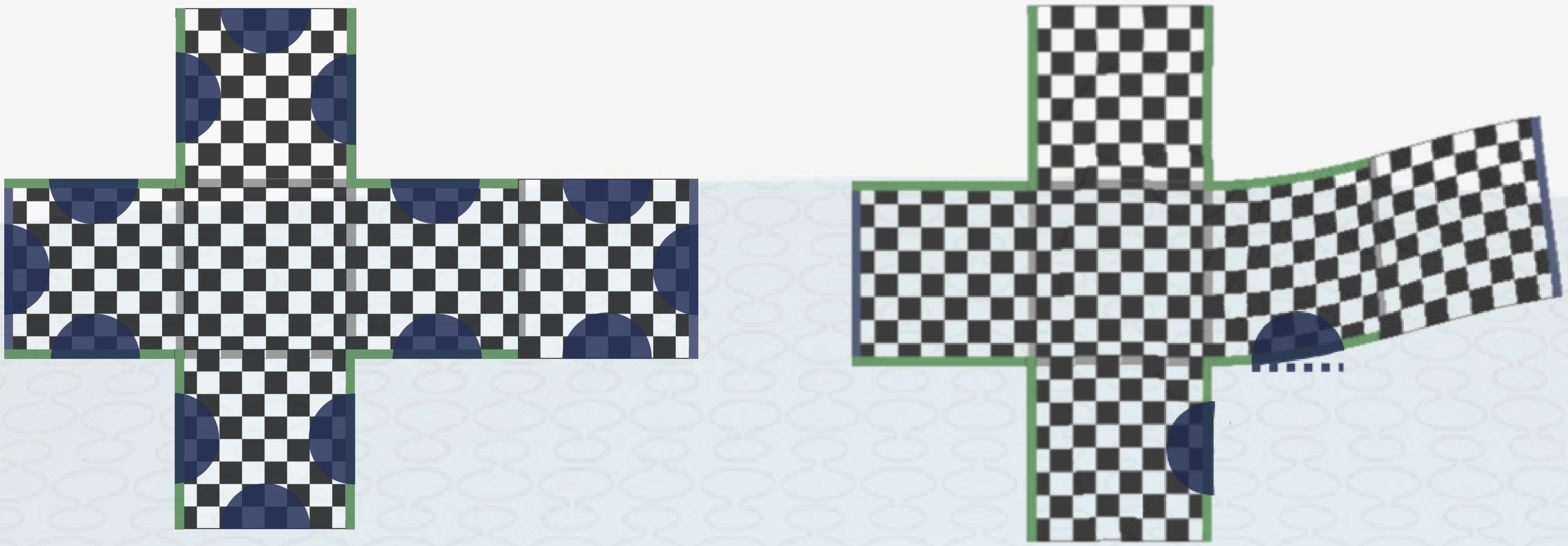
 

- Except at prescribed singularities, constant Gaussian curvature
- After *a* conformal transformation

Uniformization theorem revisited

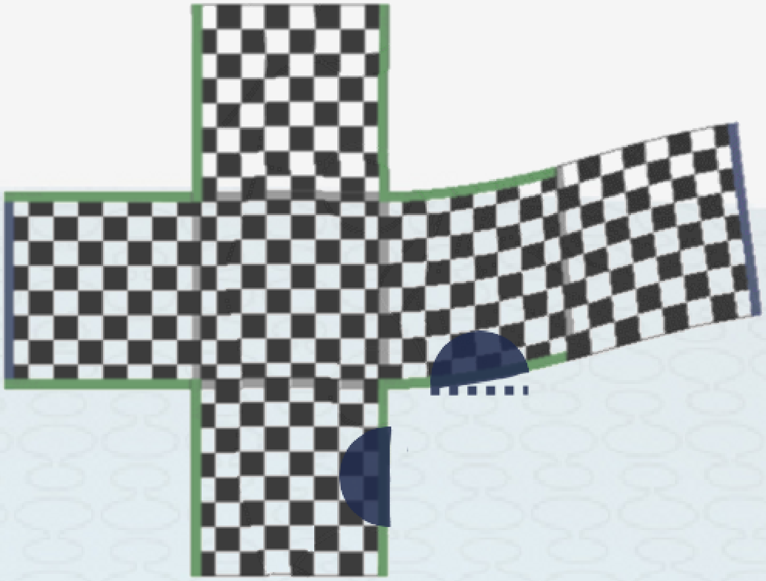
- Except at prescribed singularities, constant Gaussian curvature
- After *a* conformal transformation

Uniformization theorem revisited



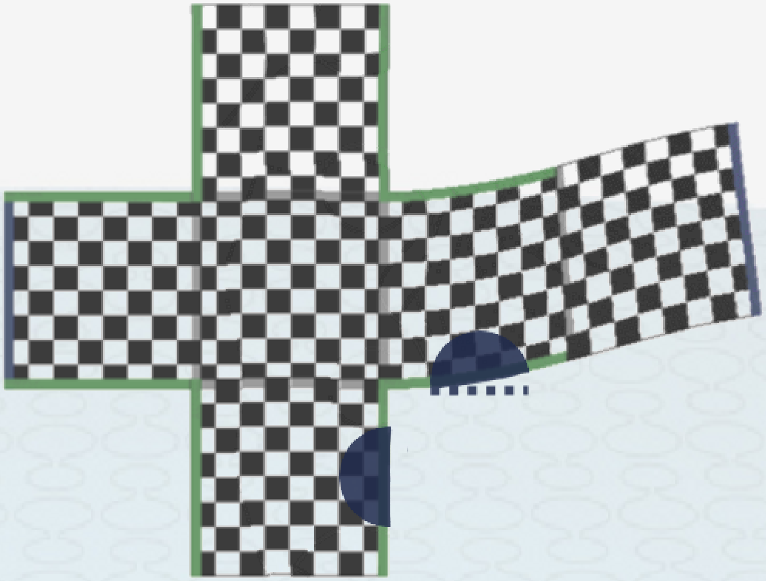
- Except at prescribed singularities, constant Gaussian curvature
- After *a* conformal transformation
  - Gaussian curvature varies from a constant
  - But still, the integral is related to the Euler characteristic

Uniformization theorem revisited



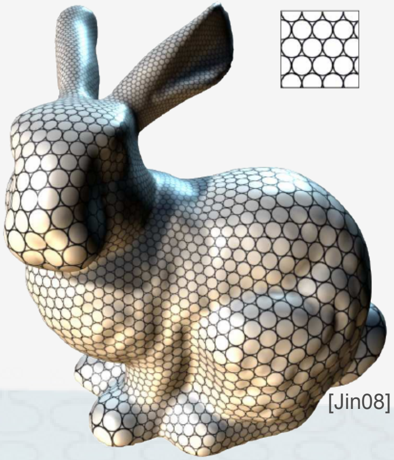
- Except at prescribed singularities, constant Gaussian curvature
- After *a* conformal transformation
  - Gaussian curvature varies from a constant
  - But still, the integral is related to the Euler characteristic
- Given 3 fixed points, there is a unique representative of conformal maps which induces constant Gaussian curvature

Uniformization theorem revisited



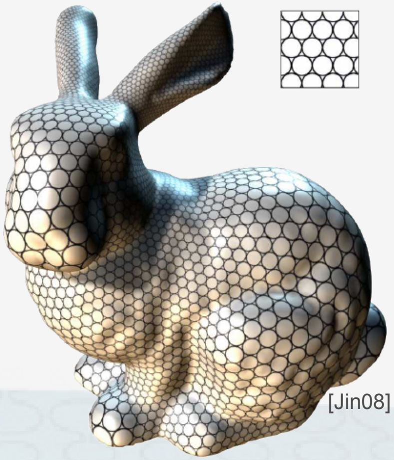
- Except at prescribed singularities, constant Gaussian curvature
- After *a* conformal transformation
  - Gaussian curvature varies from a constant
  - But still, the integral is related to the Euler characteristic
- Given 3 fixed points, there is a unique representative of conformal maps which induces constant Gaussian curvature
  - Optimization with constrained curvature (good transitions)

Optimizing metrics



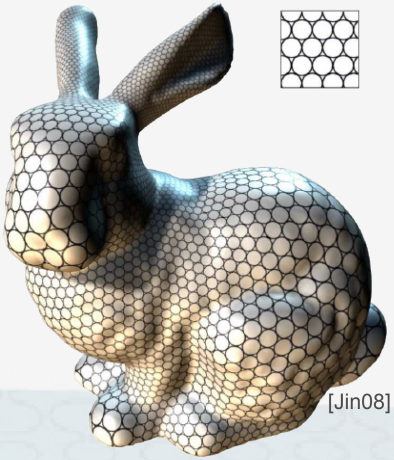
- The uniformization theorem stands originally for metrics
  - Original metric: ambient induced by the embedding in $\mathbf{R}^3$
  - $l : E \to \mathbb{R}^+$
  - Optimized metric: in the target domain (induces the unfolding)
- We need to find a formalism to conformally modify the ambient metric

Optimizing metrics



- The uniformization theorem stands originally for metrics
  - Original metric: ambient induced by the embedding in $\mathbf{R}^3$
  - $l : E \to \mathbb{R}^+$
  - Optimized metric: in the target domain (induces the unfolding)
- We need to find a formalism to conformally modify the ambient metric
  - After metric modification, I must satisfy the triangular inequality
  - After metric modification, we want circles to unfold to circles
  - Notion of circle packing metric

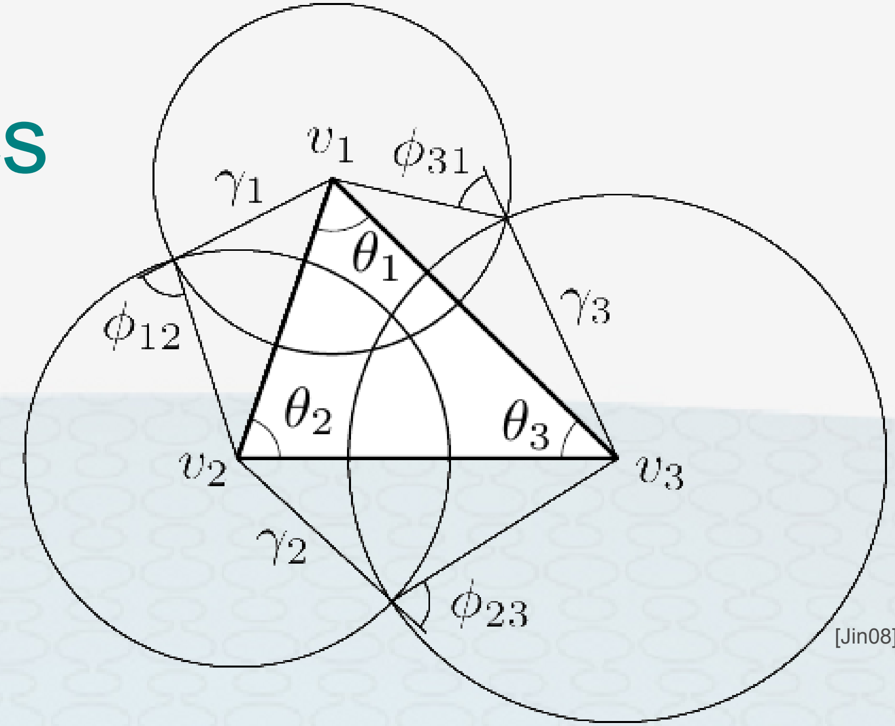
Optimizing metrics



[Jin08]

- The uniformization theorem stands originally for metrics
  - Original metric: ambient induced by the embedding in $\mathbf{R}^3$
  - $l : E \rightarrow \mathbb{R}^+$
  - Optimized metric: in the target domain (induces the unfolding)
- We need to find a formalism to conformally modify the ambient metric
  - After metric modification, I must satisfy the triangular inequality
  - After metric modification, we want circles to unfold to circles
  - Notion of circle packing metric
- Optimization process
  - Conformally optimize the abstract metric until constant curvature
  - Unfold the mesh triangle by triangle, according to the final metric

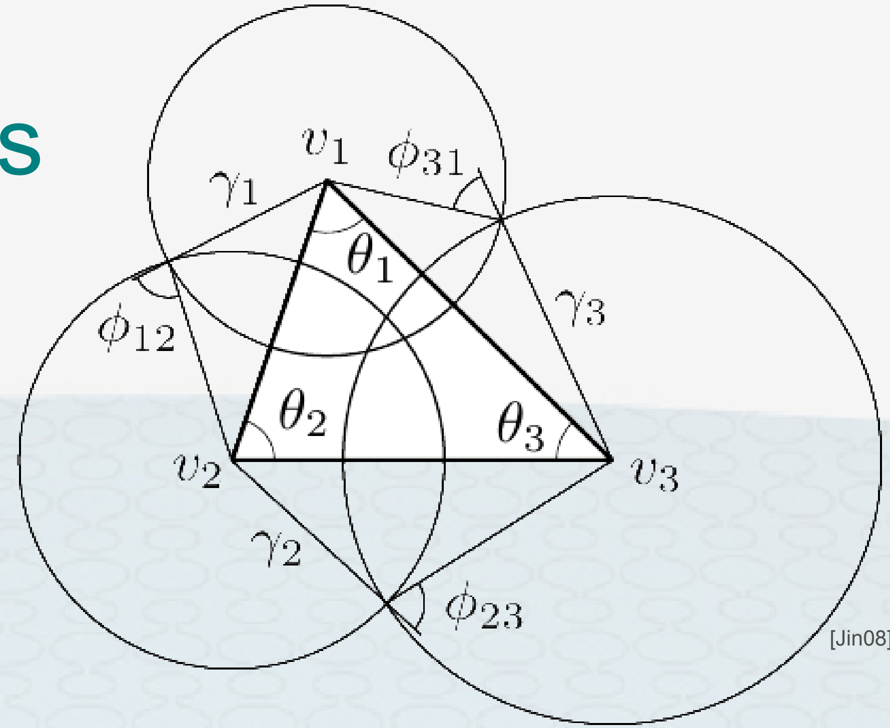
Circle packing metrics



[Jin08]

Circle packing metrics

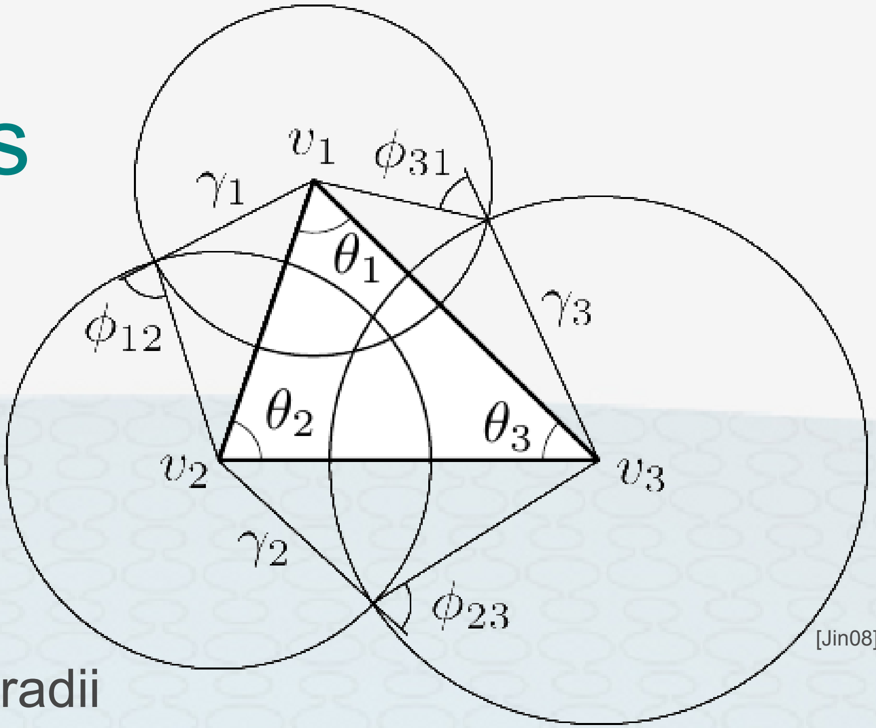
- Radius function
  - $\Gamma(v_i) = \gamma_i$
  - Radius of a circle centered at v_i



[Jin08]

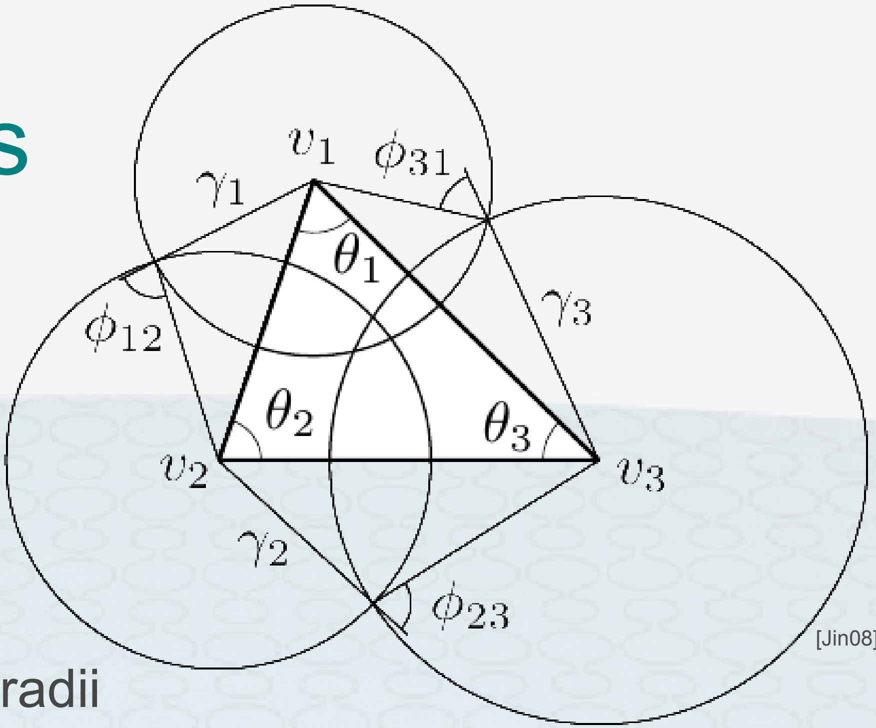
Circle packing metrics

- Radius function
  - $\Gamma(v_i) = \gamma_i$
  - Radius of a circle centered at v_i
- Angular function
  - $\theta(e_{ij}) = \alpha_{ij}$
  - Angles at the intersection of the radii



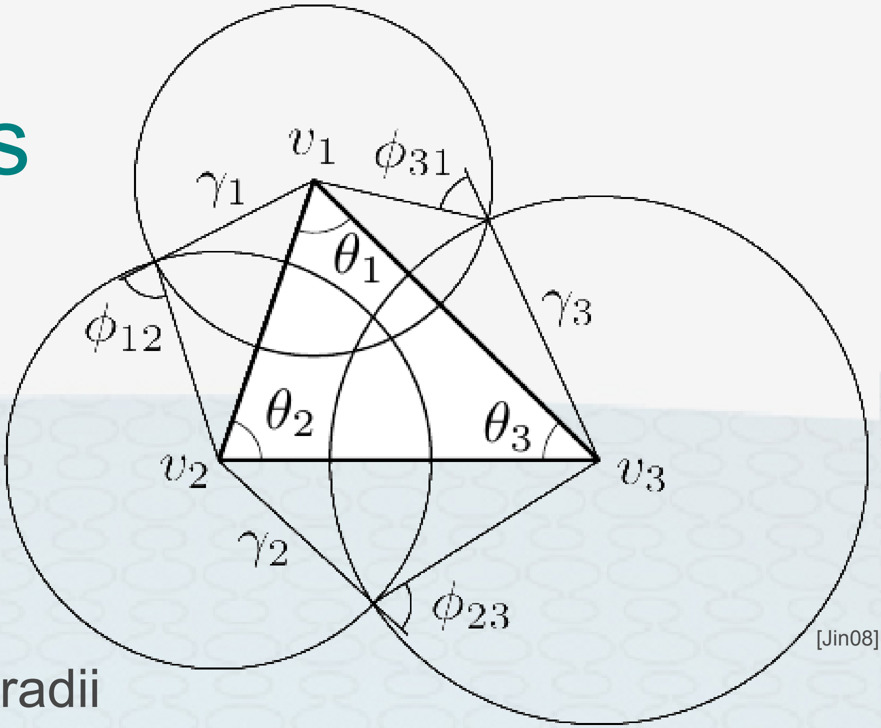
[Jin08]

Circle packing metrics



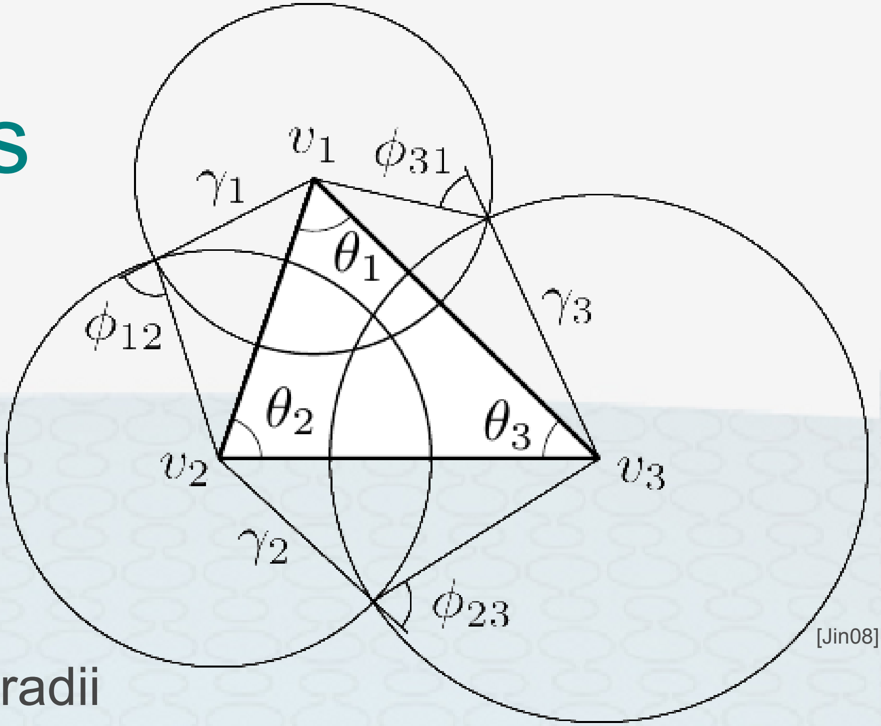
- Radius function
  - $\Gamma(v_i) = \gamma_i$
  - Radius of a circle centered at v_i
- Angular function
  - $\theta(e_{ij}) = \alpha_{ij}$
  - Angles at the intersection of the radii
- A bit of trigonometry (for euclidean targets)
  - $l_{ij}^2 = \gamma_i^2 + \gamma_j^2 + 2\gamma_i\gamma_j cos\alpha_{ij}$

Circle packing metrics



- Radius function
  - $\Gamma(v_i) = \gamma_i$
  - Radius of a circle centered at v_i
- Angular function
  - $\theta(e_{ij}) = \alpha_{ij}$
  - Angles at the intersection of the radii
- A bit of trigonometry (for euclidean targets)
  - $l_{ij}^2 = \gamma_i^2 + \gamma_j^2 + 2\gamma_i\gamma_j cos\alpha_{ij}$
- Circle packing metrics
  - (Γ, θ)
  - (Γ_1, θ_1) is conformally transformed into (Γ_2, θ_2) if $\theta_1 = \theta_2$
  - … if it preserves angles

Circle packing metrics



- Radius function
  - $\Gamma(v_i) = \gamma_i$
  - Radius of a circle centered at v_i
- Angular function
  - $\theta(e_{ij}) = \alpha_{ij}$
  - Angles at the intersection of the radii
- A bit of trigonometry (for euclidean targets)
  - $l_{ij}^2 = \gamma_i^2 + \gamma_j^2 + 2\gamma_i\gamma_j cos\alpha_{ij}$
- Circle packing metrics
  - (Γ, θ)
  - (Γ_1, θ_1) is conformally transformed into (Γ_2, θ_2) if $\theta_1 = \theta_2$
  - … if it preserves angles
  - **Optimization: radii become the variables**

Ricci flow for metric scaling

- Quick recap
  - Variables: radial component of the metric at each vertex
  - Constraints (Gauss Bonnet theorem)
    - 0 for all vertices (flat mesh)
    - Prescribed Gaussian curvature at selected vertices
  - How do we play with the variables then?

Ricci flow for metric scaling

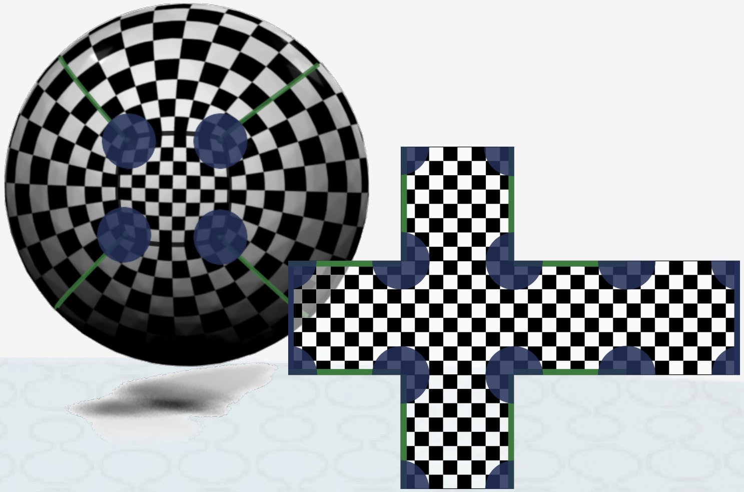
- Quick recap
  - Variables: radial component of the metric at each vertex
  - Constraints (Gauss Bonnet theorem)
    - 0 for all vertices (flat mesh)
    - Prescribed Gaussian curvature at selected vertices
  - How do we play with the variables then?

- Iterative conformal metric scaling (smooth setting)
  - $g(t) = e^{2u(t)} g(0)$, with $u(t) : \mathcal{S} \rightarrow \mathbb{R}$
  - Notion of Ricci flow: $\frac{dg(t)}{dt} = -2K(t)g(t)$
  - Flow that scales the metric according to the current curvature
  - $\frac{du(t)}{dt} = -2K(t)$

Ricci flow for metric scaling

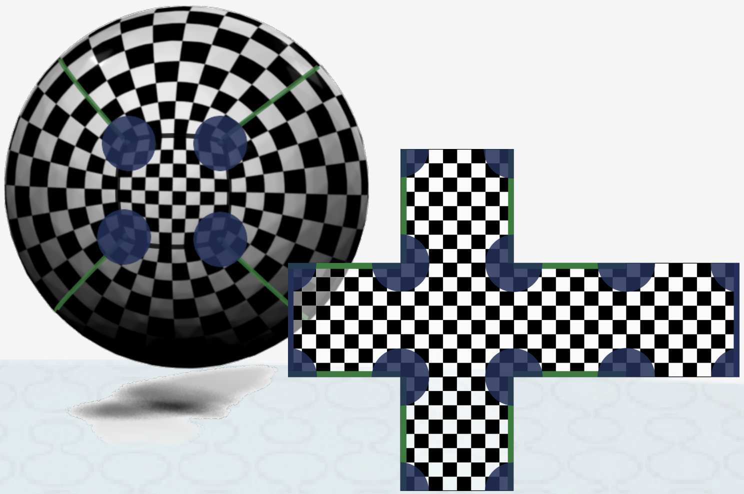
- Quick recap
  - Variables: radial component of the metric at each vertex
  - Constraints (Gauss Bonnet theorem)
    - 0 for all vertices (flat mesh)
    - Prescribed Gaussian curvature at selected vertices
  - How do we play with the variables then?
- Iterative conformal metric scaling (smooth setting)
  - $g(t) = e^{2u(t)} g(0)$, with $u(t) : \mathcal{S} \to \mathbb{R}$
  - Notion of Ricci flow: $\frac{dg(t)}{dt} = -2K(t)g(t)$
  - Flow that scales the metric according to the current curvature
  - $\frac{du(t)}{dt} = -2K(t)$
- Ricci flow and uniformization theorem [Hamilton 1988] [Chow 1991]
  - The Ricci flow converges to a metric yielding constant curvature

Discrete Ricci flow



- In short
  - Applying the Ricci flow on a circle packing metric will iteratively transform it such that it eventually yields constant curvature (0)
  - Iterative mesh unfolding according to the specified atlas layout

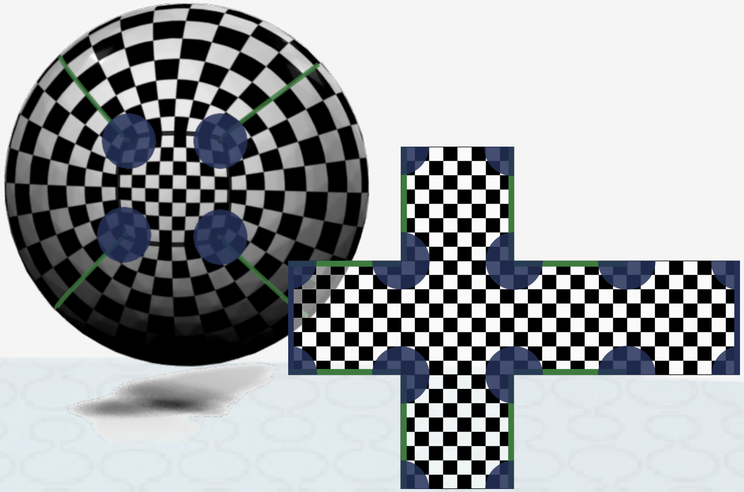
Discrete Ricci flow



- In short
  - Applying the Ricci flow on a circle packing metric will iteratively transform it such that it eventually yields constant curvature (0)
  - Iterative mesh unfolding according to the specified atlas layout

- Playing with the radial component of the metric l
  - $u_i = log\gamma_i$
  - Inserting curvature constraints K_i' in the discrete Ricci flow
    - $\frac{du_i(t)}{dt} = 2(K_i' - K_i(t))$

Discrete Ricci flow



- In short
  - Applying the Ricci flow on a circle packing metric will iteratively transform it such that it eventually yields constant curvature (0)
  - Iterative mesh unfolding according to the specified atlas layout
- Playing with the radial component of the metric l
  - $u_i = log\gamma_i$
  - Inserting curvature constraints K_i' in the discrete Ricci flow
    - $\frac{du_i(t)}{dt} = 2(K_i' - K_i(t))$
- Uniqueness of the solution [Chow and Luo 2003]
  - The discrete Ricci flow is the gradient of a convex energy

Gradient descent algorithm

Gradient descent algorithm

- Set the curvature constraint
  - 0 for all the vertices
  - Except for the singularities (determined by the atlas layout)

Gradient descent algorithm

- Set the curvature constraint
  - 0 for all the vertices
  - Except for the singularities (determined by the atlas layout)
- Set the initial γ_i and α_{ij}. Heuristic: $\gamma_i = \frac{1}{m} \sum \frac{1}{2}(l_{ki} + l_{ij} - l_{jk})$

Gradient descent algorithm

- Set the curvature constraint
  - 0 for all the vertices
  - Except for the singularities (determined by the atlas layout)
- Set the initial γ_i and α_{ij}. Heuristic: $\gamma_i = \dfrac{1}{m} \sum \dfrac{1}{2}(l_{ki} + l_{ij} - l_{jk})$

1) compute the current edge lengths l_{ij} (from γ_i and γ_j and α_{ij})

Gradient descent algorithm

- Set the curvature constraint
  - 0 for all the vertices
  - Except for the singularities (determined by the atlas layout)
- Set the initial γ_i and α_{ij}. Heuristic: $\gamma_i = \frac{1}{m} \sum \frac{1}{2}(l_{ki} + l_{ij} - l_{jk})$

1) compute the current edge lengths l_{ij} (from γ_i and γ_j and α_{ij})

2) compute the current angles of each triangle (from l_{ij})

Gradient descent algorithm

- Set the curvature constraint
  - 0 for all the vertices
  - Except for the singularities (determined by the atlas layout)
- Set the initial γ_i and α_{ij}. Heuristic: $\gamma_i = \frac{1}{m} \sum \frac{1}{2}(l_{ki} + l_{ij} - l_{jk})$

1) compute the current edge lengths l_{ij} (from γ_i and γ_j and α_{ij})
2) compute the current angles of each triangle (from l_{ij})
3) compute the current Gaussian curvature (angle deficit at each vertex)

Gradient descent algorithm

- Set the curvature constraint
  - 0 for all the vertices
  - Except for the singularities (determined by the atlas layout)
- Set the initial γ_i and α_{ij}. Heuristic: $\gamma_i = \frac{1}{m} \sum \frac{1}{2}(l_{ki} + l_{ij} - l_{jk})$

1) compute the current edge lengths l_{ij} (from γ_i and γ_j and α_{ij})
2) compute the current angles of each triangle (from l_{ij})
3) compute the current Gaussian curvature (angle deficit at each vertex)
4) update the scaling factor: $u_i \leftarrow u_i + \epsilon(K_i' - K_i)$

Gradient descent algorithm

- Set the curvature constraint
  - 0 for all the vertices
  - Except for the singularities (determined by the atlas layout)
- Set the initial γ_i and α_{ij}. Heuristic: $\gamma_i = \frac{1}{m}\sum \frac{1}{2}(l_{ki} + l_{ij} - l_{jk})$

1) compute the current edge lengths l_{ij} (from γ_i and γ_j and α_{ij})
2) compute the current angles of each triangle (from l_{ij})
3) compute the current Gaussian curvature (angle deficit at each vertex)
4) update the scaling factor: $u_i \leftarrow u_i + \epsilon(K_i' - K_i)$
5) normalize the scaling factor: $u_i \leftarrow u_i - \sum \frac{u_i}{n}$

Gradient descent algorithm

- Set the curvature constraint
  - 0 for all the vertices
  - Except for the singularities (determined by the atlas layout)
- Set the initial γ_i and α_{ij}. Heuristic: $\gamma_i = \frac{1}{m} \sum \frac{1}{2}(l_{ki} + l_{ij} - l_{jk})$

1) compute the current edge lengths l_{ij} (from γ_i and γ_j and α_{ij})
2) compute the current angles of each triangle (from l_{ij})
3) compute the current Gaussian curvature (angle deficit at each vertex)
4) update the scaling factor: $u_i \leftarrow u_i + \epsilon(K_i' - K_i)$
5) normalize the scaling factor: $u_i \leftarrow u_i - \sum \frac{u_i}{n}$
6) update the radial component of the metric (from the scaling factor)

Gradient descent algorithm

- Set the curvature constraint
  - 0 for all the vertices
  - Except for the singularities (determined by the atlas layout)
- Set the initial γ_i and α_{ij}. Heuristic: $\gamma_i = \frac{1}{m}\sum \frac{1}{2}(l_{ki} + l_{ij} - l_{jk})$

  1) compute the current edge lengths l_{ij} (from γ_i and γ_j and α_{ij})
  2) compute the current angles of each triangle (from l_{ij})
  3) compute the current Gaussian curvature (angle deficit at each vertex)
  4) update the scaling factor: $u_i \leftarrow u_i + \epsilon(K_i' - K_i)$
  5) normalize the scaling factor: $u_i \leftarrow u_i - \sum \frac{u_i}{n}$
  6) update the radial component of the metric (from the scaling factor)

- Repeat until the curvature is close to the constraint value (threshold)

Gradient descent algorithm

- Set the curvature constraint
  - 0 for all the vertices
  - Except for the singularities (determined by the atlas layout)
- Set the initial γ_i and α_{ij}. Heuristic: $\gamma_i = \frac{1}{m} \sum \frac{1}{2}(l_{ki} + l_{ij} - l_{jk})$

1) compute the current edge lengths l_{ij} (from γ_i and γ_j and α_{ij})
2) compute the current angles of each triangle (from l_{ij})
3) compute the current Gaussian curvature (angle deficit at each vertex)
4) update the scaling factor: $u_i \leftarrow u_i + \epsilon(K_i' - K_i)$
5) normalize the scaling factor: $u_i \leftarrow u_i - \sum \frac{u_i}{n}$
6) update the radial component of the metric (from the scaling factor)

- Repeat until the curvature is close to the constraint value (threshold)
- Pin one triangle in the plane, iteratively pin neighbors (from the metric)

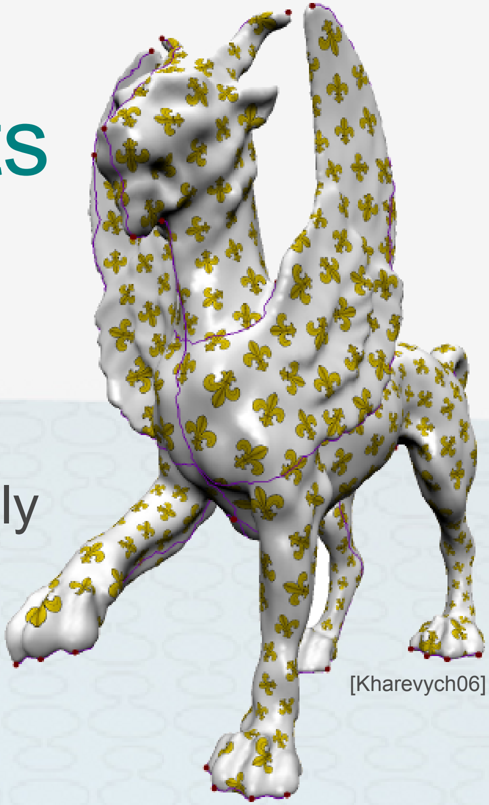
Gradient descent algorithm

- Set the curvature constraint
  - 0 for all the vertices
  - Except for the singularities (determined by the atlas layout)
- Set the initial γ_i and α_{ij}. Heuristic: $\gamma_i = \frac{1}{m} \sum \frac{1}{2}(l_{ki} + l_{ij} - l_{jk})$

  1) compute the current edge lengths l_{ij} (from γ_i and γ_j and α_{ij})
  2) compute the current angles of each triangle (from l_{ij})
  3) compute the current Gaussian curvature (angle deficit at each vertex)
  4) update the scaling factor: $u_i \leftarrow u_i + \epsilon(K_i' - K_i)$
  5) normalize the scaling factor: $u_i \leftarrow u_i - \sum \frac{u_i}{n}$
  6) update the radial component of the metric (from the scaling factor)

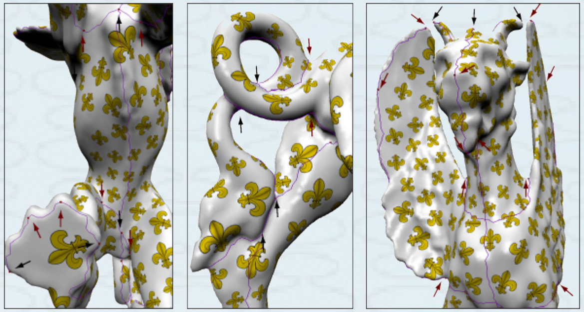
- Repeat until the curvature is close to the constraint value (threshold)
- Pin one triangle in the plane, iteratively pin neighbors (from the metric)
- Simple, right?

Chronology and enhancements



[Kharevych06]

- Three different techniques appeared ~ simultaneously
  - Different formulations
  - In essence, exactly the same process
  - [Kharevych et al. 2006]
  - **[Jin et al. 2008]**
    - Connection to Ricci flow
    - Euclidean, spherical and hyperbolic targets
    - Gradient descent and Newton's method
  - [Ben Chen et al. 2008]
    - Non iterative approach
    - Automatic singularity layout



Are we done now?!

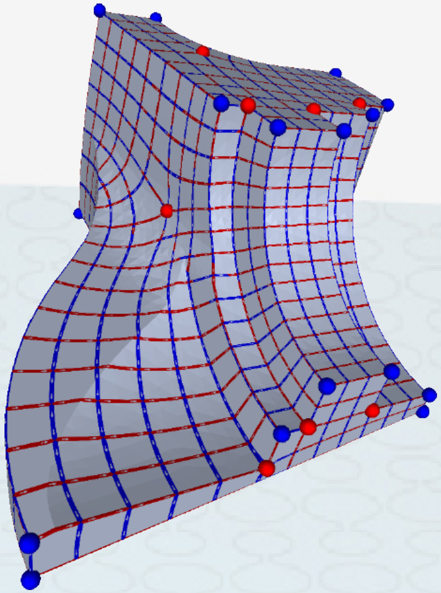
[Jin08]

- There is still room for enhancements
  - Generation of the initial atlas layout?
  - Control on the singularities
  - Control on the alignment and orientation
- Towards artistic quadrangulation
- How about 3-manifolds?

Notion of surface quadrangulation

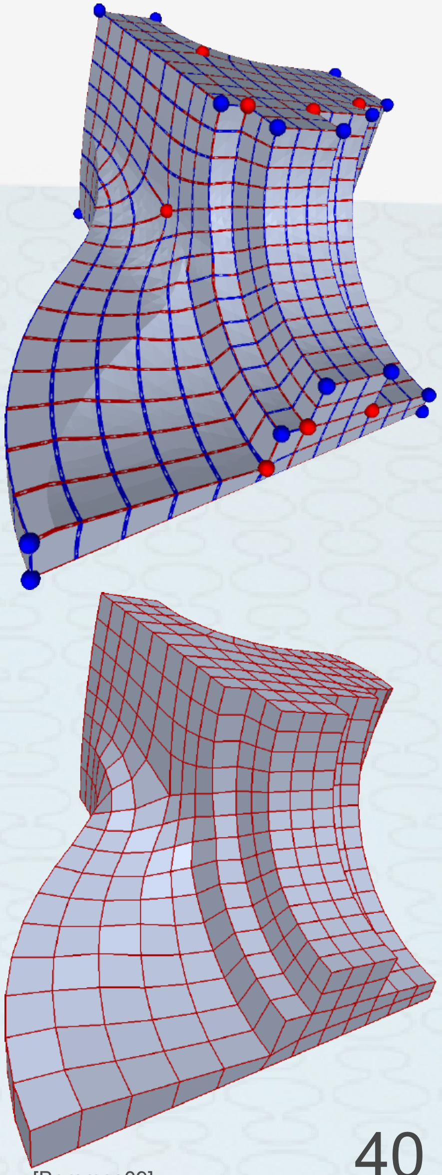
Notion of surface quadrangulation

- Given a global parameterization
  - Iso-contouring of the (u, v) fields
  - Yields a surface discretization made of quadrilaterals



Notion of surface quadrangulation

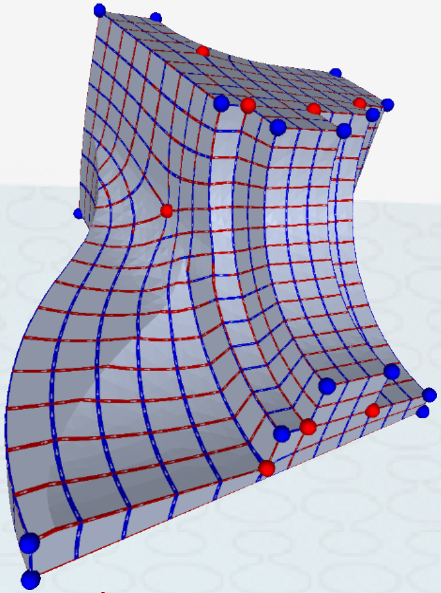
- Given a global parameterization
  - Iso-contouring of the (u, v) fields
  - Yields a surface discretization made of quadrilaterals

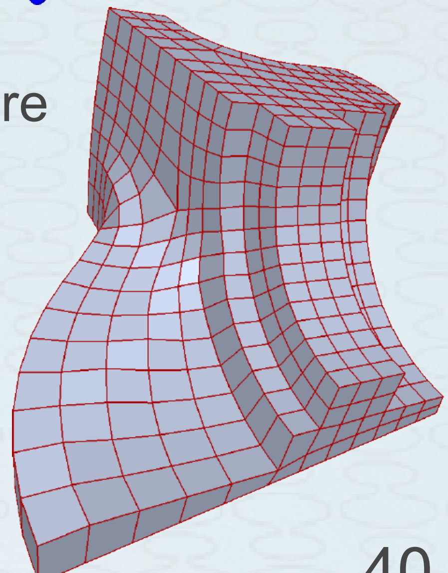


[Bommes09]

Notion of surface quadrangulation

- Given a global parameterization
  - Iso-contouring of the (u, v) fields
  - Yields a surface discretization made of quadrilaterals
- Why is it interesting?
  - The discretization *captures* the unfolding
    - Pin one quad and continue (instant unfolding)
  - The discretization captures the geometrical structure
    - Numerical stability (animation, simulation, etc.)

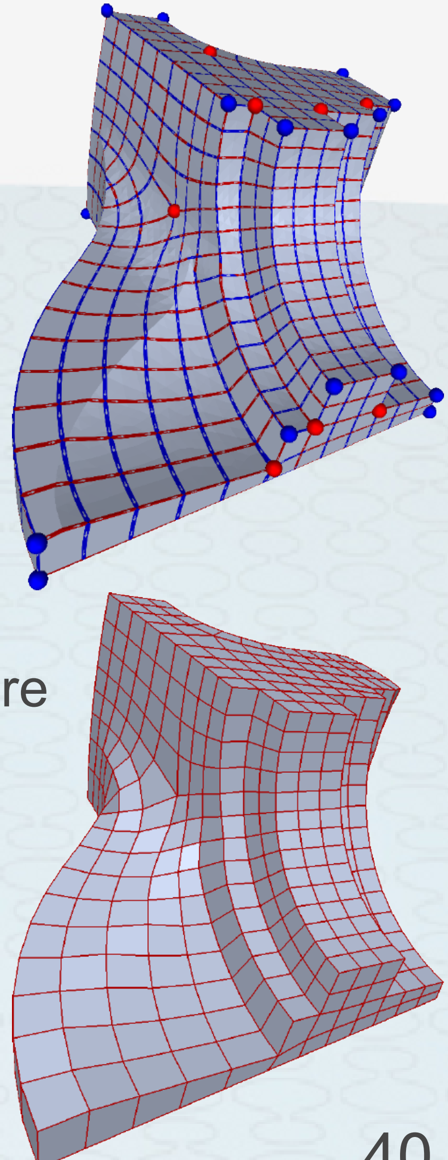




[Bommes09]

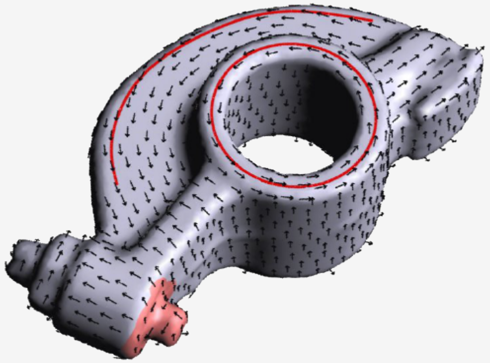
Notion of surface quadrangulation

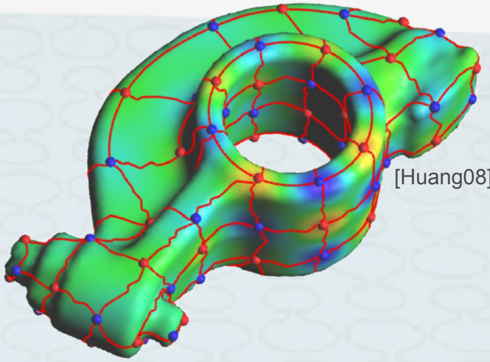
- Given a global parameterization
  - Iso-contouring of the (u, v) fields
  - Yields a surface discretization made of quadrilaterals
- Why is it interesting?
  - The discretization *captures* the unfolding
    - Pin one quad and continue (instant unfolding)
  - The discretization captures the geometrical structure
    - Numerical stability (animation, simulation, etc.)
- From a reverse engineering point of view (**jpg** → **svg**)
  - Artists generate surfaces made of quads
  - Quadrangulating a surface makes it ready for the geometric modeling pipeline



[Bommes09]

Initial altas layout





- Defines the extraordinary vertices of the mesh
- Defines the orientation and alignment of the edges

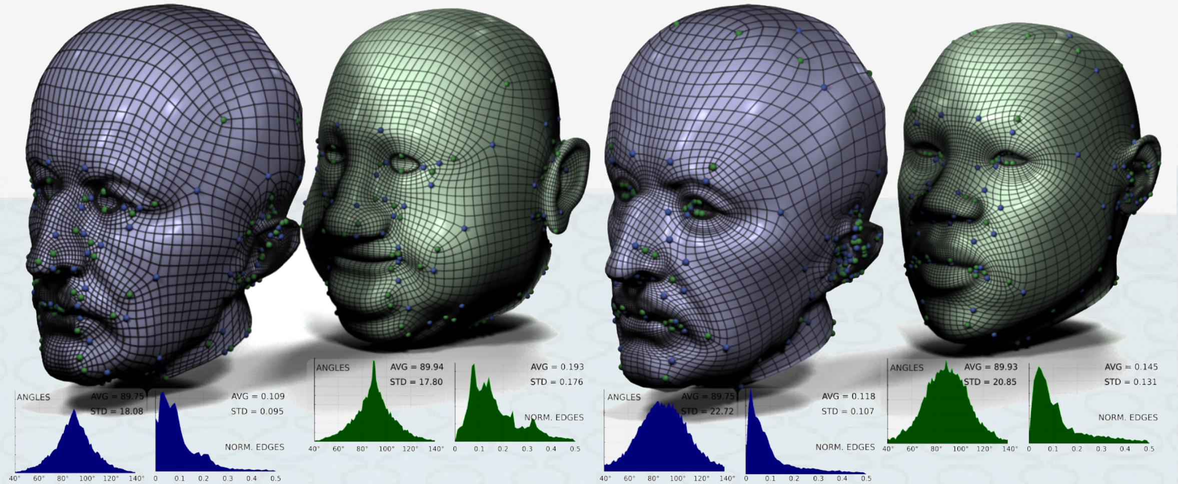
- Fully manual
  - Singularity graph [Tong et al. 2006]
  - Polycube maps [Tarini et al. 2004]

- Fully automatic
  - Based on the Morse-Smale complex [Dong et al. 2006]

- Semi-automatic
  - Driven by sparse directional constraints
  - [Huang et al. 2008], **[Bommes et al. 2009]**

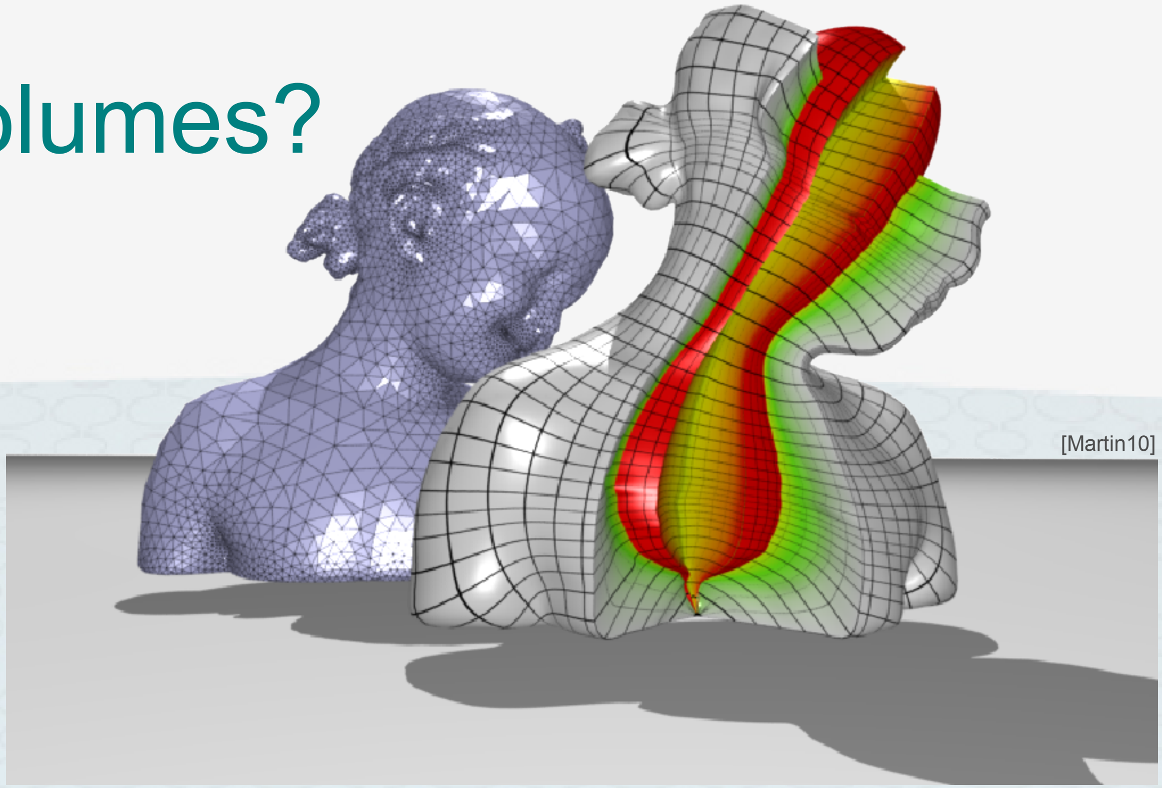
- Then, you could use any global parameterization technique (in theory)

Towards artistic quadrangulation



- Automatic techniques?
  - Artists say “no way!”
    - Semantic of the surface (not necessarily related to its geometry)
    - Need for control
- User driven techniques
  - Still require a lot of intervention (plus advanced skills)
- More general problem of cross parameterization
  - $\psi : \mathcal{S}_1 \rightarrow \mathcal{S}_2$
  - Applications in shape registration, recognition, etc.

What about volumes?



[Martin10]

- For the same reasons, interesting to reverse engineer too
  - Harmonic maps with prescribed singularities
  - [Martin et al. 2009], [Martin et al. 2010]
  - [Xia et al. 2010]
- Discrete Ricci flow on PL 3-manifolds?
- Still a lot to do :)

A few useful references

- "*Surface Parameterization : A tutorial and a survey*", Floater M. and Hormann K., Advances in Multiresolution for Geometric Modelling, pp. 157-186, 2005.
- "*Mesh Parameterization: Theory and Practice*", Hormann K., Levy B., Sheffer A., ACM SIGGRAPH Course Notes, 2007
- "*Least Squares Conformal Maps*", Levy B., Petitjean S., Ray N., Maillot J., Proc. Of ACM SIGGRAPH 2002.
- "*Discrete Surface Ricci Flow*", Lin M., Kim J., Luo F., Gu X., IEEE Transactions on Visualization and Computer Graphics 2008.