Platonic Holo-Moduli: a constructive 6D code for sphere and the five regular solids

Douglas C. Youvan

doug@youvan.com

www.youvan.ai

October 30, 2025

We present a constructive six dimensional nonnegative coordinate space that encodes the geometry, up to similarity, of six canonical shape classes: sphere, tetrahedron, cube, octahedron, dodecahedron, and icosahedron. Each axis carries exactly one intrinsic degree of freedom, its scale, so a single nonzero coordinate fixes all metric properties of the corresponding class: surface area, volume, inradius, circumradius, and dihedral or curvature quantities. The space projectivizes to the 5 simplex by normalizing total scale, separating type mixture from global size. On this space we define a linear holographic multiplex: a single complex field encodes any 6 vector via fixed carrier codewords, and exact recovery follows by inner products when carriers are orthonormal, with graceful degradation governed by the Gram matrix when they are merely low coherence. The sphere axis enables an elegant circumsphere constraint manifold: choosing a common radius determines the compatible edge length for each regular solid by closed form constants, yielding a one parameter family that intersects each one hot ray at a unique point. We analyze symmetry actions, topology as a fan of six rays with a singular apex, information geometry on the simplex, robustness under noise, and software or optical implementations. The framework generalizes to finite shape alphabets while preserving orthogonal, interpretable coordinates.

Keywords: sphere, Platonic solids, moduli space, holography, multiplexing, simplex geometry, duality, matched filtering, shape signatures, circumsphere constraint.

A collaboration with GPT-5. 41 pages.

Outline

1. Background and motivation

1.1 Regular geometries up to similarity

Regular shapes have maximal symmetry and minimal internal freedom. Once translation and rotation are removed, a sphere, tetrahedron, cube, octahedron, dodecahedron, or icosahedron is determined by a single continuous parameter: its scale. All other metric quantities are fixed multiples of that scale. For example, when edge length is used as the scale for a regular solid, surface area and volume are proportional to the square and cube of that length with fixed algebraic constants; dihedral angles are fixed numbers independent of scale. For the sphere, radius alone sets surface area 4 pi R squared, volume 4 pi R cubed over 3, mean curvature 2 over R, and Gaussian curvature 1 over R squared. This collapse of geometry to one degree of freedom per class is precisely what makes a small, interpretable coordinate code possible. The six selected classes also form a natural canonical basis for basic 3D geometry: one perfectly curved object and the five perfectly regular polyhedra. Treating these as similarity classes focuses on intrinsic geometry rather than pose, which can be handled separately if needed.

1.2 A six axis moduli code and its use cases

The six axis code is simply R nonnegative to the sixth power with one coordinate per class: sphere, tetrahedron, cube, octahedron, dodecahedron, icosahedron. Each coordinate stores the scale for a fixed canonical template of its class. Reading a coordinate back gives complete metric information up to similarity for that class; combinatorial data such as numbers of faces, edges, and vertices are constant within the class.

This compact code is useful in several ways:

- Geometric catalogs and teaching: a single vector selects a class and fixes all of its metric properties, supporting interactive visualizations and closed form computations.
- Model selection and detection: in graphics or vision, a regressor can map data to the six axis code and immediately report which class is present and at what scale, optionally with a one hot constraint.
- Holographic dictionaries: the code can be embedded into one complex field that is easy to store, transmit, or display optically, yet decodes back to the six numbers exactly.
- Common circumsphere scenes: adding the constraint that all solids are inscribed in a sphere with radius equal to the sphere coordinate turns the code into a one parameter family along each ray with closed form relations between radius and edge length. This is a natural way to tie multiple classes together without adding degrees of freedom.

- Compression and indexing: large libraries of clean geometric exemplars can be summarized by small histograms on the 5 simplex plus an overall scale, enabling fast search and clustering.

Because the coordinates are interpretable and disentangled by construction, the code supports sparse penalties, probabilistic mixtures on the normalized simplex, and simple rules for selection and thresholding.

1.3 Constructive holography as linear coding

Constructive holography here means a linear map that multiplexes the six coordinates into a single object together with a matched linear demultiplexer that recovers them. Pick any complex inner product space for the carrier field, such as complex images for computer generated holography or abstract functions for a software signature. Choose six unit norm codewords, one per class, designed to be orthonormal or at least low coherence. The encoder is the weighted sum of these codewords with weights equal to the six coordinates. The decoder computes inner products with the same codewords. If the codewords are orthonormal, decoding returns the exact coordinates. If they are merely low coherence, the Gram matrix gives a closed form linear inverse and bounds the effect of noise through its condition number. This is classical linear coding and matched filtering, not a conjectural principle.

In optics, orthogonality can be achieved by assigning distinct Fourier tilts, polarizations, wavelengths, or coded phases. In pure software, one can use spherical harmonic shape signatures or signed distance fields with orthogonal phase masks. The same linear scheme supports mixtures on the normalized 5 simplex by interpreting coordinates as nonnegative weights that sum to one, plus a separate global scale. Because everything is linear and explicit, the system is easy to implement, analyze, and extend: carriers can be designed to minimize cross talk, noise performance follows standard signal models, and decoding is a single set of inner products or an FFT crop when tilt multiplexing is used.

2. The 6D moduli cone

2.1 Definition: R nonnegative to the sixth power with axes S, T, C, O, D, I

We define the code space
X = R_{>=0}^6 = {(a_S, a_T, a_C, a_O, a_D, a_I) : each a_k >= 0},
with one coordinate per similarity class:

- S = sphere
- T = tetrahedron

- C = cube
- O = octahedron
- D = dodecahedron
- I = icosahedron

Each axis carries a single geometric degree of freedom: scale. Internally we fix canonical, centered, unit templates for each class; the coordinate a_k is the scale applied to that template. Because the classes are regular, all metric properties are fixed functions of a_k alone. X is a closed convex cone in R^6. It is contractible, has a linear structure for optimization, and supports both Euclidean and log Euclidean coordinates ($u_k = \log a_k$ for $a_k > 0$) when multiplicative behavior is desirable.

Symmetry actions are simple. Relabeling classes acts by coordinate permutation (the S6 group). Duality acts by the permutation that swaps C with O and D with I; S and T are fixed points of this involution.

2.2 One hot fan: six rays and the apex

Many tasks require selecting exactly one class at a time. The corresponding subspace is the one hot fan
F = union over k in {S,T,C,O,D,I} of { $a_k > 0$ and $a_j = 0$ for $j \neq k$ }.
Topologically, F is a star of six open rays meeting at the origin. The origin is the apex (a singular 0 stratum); each open ray is a 1 stratum on which all combinatorial invariants are constant and all metric invariants vary smoothly with a_k. Removing the apex disconnects F into six components, one per class.

This stratified view is practical:

- encoders and decoders are block diagonal by construction (no cross talk between axes);
- selection can be encouraged with L1 penalties on X, which promote sparsity and recover one hot solutions under standard conditions;
- algorithms that depend on ratios or logs should avoid the apex or work on the normalized model below.

2.3 Projectivization to the 5 simplex and separation of scale

To separate type mixture from global size, normalize by total scale. For any nonzero a in X, define
$s = \sum_k a_k$ (global scale, $s > 0$)
$w_k = a_k / s$ (mixture weights, w in Δ^5 and $\sum_k w_k = 1$).

This yields a product decomposition
X \ {0} isomorphic to (Delta^5) x (0, infinity),
where Delta^5 is the 5 simplex of nonnegative 6 tuples that sum to 1. Intuitively:

- w encodes which class or mixture of classes you have (pure one hot states are the six vertices of Delta^5; convex combinations are interior points);
- s encodes how big the overall configuration is.

Benefits of the simplex view:

- clear probabilistic semantics for mixtures (w as categorical weights);
- natural metrics: L2 or L1 on the simplex for geometry, Fisher information if w is treated as a distribution;
- stable numerics: the apex is excluded by fixing s, while w remains bounded inside Delta^5.

Common operations become simple:

- one hot selection = choose a vertex of Delta^5;
- soft selection = choose a point near a vertex;
- scale changes = move along the (0, infinity) factor with w fixed;
- circumsphere constraints can be expressed as linear maps from s to class scales through fixed constants, applied after choosing a vertex.

In practice, many pipelines use the pair (w, s) as the primary state. Encoders map (w, s) to a by a = s w. Decoders recover a, then compute s and w by the formulas above. This separation keeps type decisions and size decisions clean, interpretable, and numerically well conditioned.

3. Geometry from one coordinate per class

3.1 Completeness up to similarity

Each axis stores one scalar scale for a fixed canonical template of its class: sphere (radius), or a regular solid (edge length, or any equivalent scale). Because these shapes are regular, all metric quantities are fixed functions of that single scale. Surface area scales as scale^2, volume as scale^3, and all angles are constant. Thus one coordinate per class is sufficient to recover every metric property up to similarity. Pose (translation and rotation) is excluded by design and can be handled separately if ever needed.

Key facts:

- Homogeneity: if you multiply the class scale by s, then all lengths multiply by s, areas by s^2, volumes by s^3.
- Angles and combinatorial data do not change with scale.

3.2 Closed form metric properties for sphere and solids

Let a be edge length for a regular solid, and let R be sphere radius. Let phi = (1 + sqrt(5)) / 2. Below are standard formulas you can use directly.

Sphere (radius R)

- Surface area: 4 pi R^2
- Volume: (4 pi / 3) R^3
- Mean curvature: 2 / R
- Gaussian curvature: 1 / R^2

Tetrahedron

- Faces, edges, vertices: F=4, E=6, V=4
- Surface area: sqrt(3) a^2
- Volume: a^3 / (6 sqrt(2))
- Circumradius: R = (sqrt(6) / 4) a
- Inradius: r = (sqrt(6) / 12) a
- Dihedral angle: arccos(1/3) about 70.5288 deg

Cube

- F=6, E=12, V=8
- Surface area: 6 a^2
- Volume: a^3
- Circumradius: R = (sqrt(3) / 2) a
- Inradius: r = (1 / 2) a
- Dihedral angle: 90 deg

Octahedron

- F=8, E=12, V=6
- Surface area: 2 sqrt(3) a^2
- Volume: (sqrt(2) / 3) a^3
- Circumradius: R = (sqrt(2) / 2) a
- Inradius: r = (sqrt(6) / 6) a
- Dihedral angle: arccos(-1/3) about 109.4712 deg

Dodecahedron

- F=12, E=30, V=20
- Surface area: 3 sqrt(25 + 10 sqrt(5)) a^2
- Volume: ((15 + 7 sqrt(5)) / 4) a^3
- Circumradius: R = (sqrt(3) / 4) (1 + sqrt(5)) a = (sqrt(3) phi / 2) a
- Inradius: r = (1 / 20) sqrt(250 + 110 sqrt(5)) a
- Dihedral angle: arccos(-sqrt(5) / 5) about 116.565 deg

Icosahedron

- F=20, E=30, V=12
- Surface area: 5 sqrt(3) a^2
- Volume: (5 (3 + sqrt(5)) / 12) a^3
- Circumradius: R = (1 / 4) sqrt(10 + 2 sqrt(5)) a
- Inradius: r = (sqrt(3) / 12) (3 + sqrt(5)) a
- Dihedral angle: about 138.1897 deg

Notes

- Duality cross-checks: cube and octa are dual; dodeca and icosa are dual. Known identities between radii and face geometry let you sanity-check implementations.
- Any other metric (edge-to-edge distances, face heights, solid angles at vertices) follows from the same single scale a and these constants.

3.3 Piecewise constant combinatorics along rays

On each open ray of the one hot fan (one class active, scale > 0), the combinatorial type is fixed: the face/edge/vertex counts and incidence relations do not change with scale. Crossing the apex to a different ray switches to a different combinatorial type.

Combinatorics summary

- Sphere: no polygonal combinatorics in the usual sense; treat as a smooth class.
- Tetrahedron: F=4, E=6, V=4
- Cube: F=6, E=12, V=8
- Octahedron: F=8, E=12, V=6
- Dodecahedron: F=12, E=30, V=20
- Icosahedron: F=20, E=30, V=12

Implication

- Any function that depends only on combinatorics (for example Euler characteristic V - E + F) is locally constant on each ray. Functions that depend on metric data vary smoothly with the single scale parameter for that ray.

4. Common circumsphere constraint manifold

4.1 Edge length to shared radius via fixed constants

Impose a single circumsphere of radius R shared by whichever regular solid is “active.” For each class k, the circumradius R and edge length a_k satisfy R = c_k a_k with a fixed constant c_k. Equivalently a_k = R / c_k.

Canonical constants (edge length a, sphere radius R):

- Tetrahedron: c_T = sqrt(6) / 4 so a_T = 4 R / sqrt(6)
- Cube: c_C = sqrt(3) / 2 so a_C = 2 R / sqrt(3)
- Octahedron: c_O = sqrt(2) / 2 so a_O = sqrt(2) R
- Dodecahedron: c_D = (sqrt(3) / 4) (1 + sqrt(5)) so a_D = 4 R / (sqrt(3) (1 + sqrt(5)))
- Icosahedron: c_I = (1 / 4) sqrt(10 + 2 sqrt(5)) so a_I = 4 R / sqrt(10 + 2 sqrt(5))
- Sphere: coordinate is the radius itself, a_S = R

These are the only relations you need to tie the “sphere axis” to any one of the five regular solids at the same circumsphere.

4.2 Parametric description and intersections with one hot rays

Work in the 6D code a = (a_S, a_T, a_C, a_O, a_D, a_I).

For each class k in {T, C, O, D, I} define a 1D curve gamma_k(R) that lives in the 2 coordinate plane spanned by the sphere axis S and the class axis k:

- gamma_T(R) = (R, 4R/sqrt(6), 0, 0, 0, 0)
- gamma_C(R) = (R, 0, 2R/sqrt(3), 0, 0, 0)
- gamma_O(R) = (R, 0, 0, sqrt(2) R, 0, 0)
- gamma_D(R) = (R, 0, 0, 0, 4R/(sqrt(3)(1+sqrt(5))), 0)
- gamma_I(R) = (R, 0, 0, 0, 0, 4R/sqrt(10+2sqrt(5)))

Each gamma_k is a ray parameterized by the shared radius R > 0. The full circumsphere manifold is the union of these five rays. Geometrically, for each k you are “two sparse”: only the sphere and the chosen solid are nonzero, and their scales are rigidly linked by the constant c_k.

Simplex view (separating type from size). Normalize by total scale s = sum_i a_i and set weights w_i = a_i / s. On gamma_k the weight vector w lies on the 1D edge of the 5 simplex connecting vertex S and vertex k. The ratio along that edge is fixed by
w_k / w_S = a_k / a_S = (R / c_k) / R = 1 / c_k,
while s scales freely with R. Thus:

- choosing k picks an edge of the simplex,
- the mixture proportion on that edge is the constant 1 / c_k up to normalization,
- the overall size lives in the separate scale factor s.

Intersections with one hot rays. The one hot fan for the solids alone (exactly one of T, C, O, D, I nonzero, sphere ignored) is 5 disjoint rays. Each gamma_k intersects the k ray at every R if you allow the sphere axis to be nonzero; if you insist on literal one hot including the sphere, then the intersection is only at the apex R = 0. In practice, most pipelines treat the sphere axis as a coupled companion coordinate, so the “active” state is the 2D plane spanned by S and k, traced along gamma_k.

4.3 Consistency checks using dual pairs

Dual pairs are cube with octa and dodeca with icosa. On a shared circumsphere they exhibit clean, checkable relations.

Same inradius under a common circumsphere.

- Cube and octa with the same R have equal inradii:
  $r_C = a_C / 2 = (2R/\sqrt{3}) / 2 = R / \sqrt{3}$
  $r_O = (\sqrt{6}/6)\, a_O = (\sqrt{6}/6)(\sqrt{2}\, R) = R / \sqrt{3}$
- Dodeca and icosa with the same R also have equal inradii:
  $r_D = k_D R$, $r_I = k_I R$ with $k_D = k_I \approx 0.7946544723$
  (either formula from Section 3.2 yields this number when substituted with $a = R / c$)

Edge length ratios for duals on the same R.

- Octa to cube: $a_O / a_C = (\sqrt{2}\, R) / (2R/\sqrt{3}) = (\sqrt{6}) / 2 \approx 1.2247448714$
- Icosa to dodeca: $a_I / a_D = [4 / \sqrt{10 + 2\sqrt{5}}] / [4 / (\sqrt{3}(1 + \sqrt{5}))]$
  which simplifies numerically to ≈ 1.4733704196

Self dual tetra sanity check.

- Tetra on a sphere of radius R has inradius $r_T = (\sqrt{6}/12)\, a_T = (\sqrt{6}/12)(4R/\sqrt{6})$
  $= R / 3$

These equalities are robust unit tests for any implementation of the circumsphere manifold: given R, compute a_k from 4.1, derive r and other radii from Section 3.2, and verify that dual-pair inradii match and that edge-length ratios take the listed values.

5. Platonic holo multiplex in six channels

5.1 Encoder as linear superposition, decoder as inner products

Set up a complex inner product space H for carrier fields. Examples: complex hologram images on a finite grid with the usual dot product, or complex valued functions with an L2 style inner product. Choose six carrier codewords $h_S, h_T, h_C, h_O, h_D, h_I$ in H, one per axis, each normalized to unit norm.

Given a code vector a in R nonnegative to the sixth power, the encoder forms a single field
$E(a) = a_S h_S + a_T h_T + a_C h_C + a_O h_O + a_D h_D + a_I h_I$.

The simplest decoder computes inner products with the same codewords
y_k = inner product of E(a) and h_k, for k in {S, T, C, O, D, I}.
Stacking these six numbers gives D naive(E(a)).

Interpretation

- If the codewords are orthonormal, then y_k equals a_k exactly.
- If not, y = G a where G is the 6 by 6 Gram matrix with entries G_{ij} = inner product of h_i and h_j. In that case a is recovered by solving G a = y.

This is transparent linear coding: the encoder is a synthesis operator, the decoder is the matched analysis operator.

Practical carrier choices

- Fourier or tilt multiplexing: $h_k(x) = U_k(x)$ times exp(i 2 pi k_k dot x), with carrier frequencies k_k spaced so that their Fourier lobes do not overlap on the sensor.
- Polarization multiplexing: assign six Jones vectors with low mutual coherence and decode with polarization analysis.
- Wavelength multiplexing: assign six wavelengths if hardware allows; decode by spectral splitting.
- Phase code multiplexing: multiply each U_k by a different binary or ternary mask with near orthogonality; decode by correlation.

Pure software variant

- Let U_k be a stable shape signature for the unit template of class k, for example a spherical harmonic signature or a signed distance field. Bind an orthogonal phase mask to each U_k to create the codewords h_k.

5.2 Exact recovery with orthonormal carriers

Assume the six codewords form an orthonormal set in H. Then
inner product of h_i and h_j equals 1 if i equals j, and 0 otherwise.

For any a in R nonnegative to the sixth power,
D naive(E(a)) returns the six tuple of inner products
(inner product of E(a) with h_S, inner product of E(a) with h_T, ...) equals a.

Hence D naive composed with E is the identity on the code space. Linearity gives immediate superposition: sums of code vectors map to sums of fields, and the same inner products separate them perfectly.

Noise and perturbations
Let the observed field be y = E(a) + n with noise n in H. The estimate is
a hat_k = inner product of y and h_k equals a_k plus inner product of n and h_k.
If the noise is white with variance sigma squared per dimension, each channel sees independent Gaussian error with variance sigma squared. Signal to noise grows with total exposure or aperture area in the usual way.

FFT demultiplex shortcut for tilt carriers

- If h_k are built as unit templates modulated by distinct Fourier tilts, computing the 2D FFT of the hologram places each channel's energy in disjoint frequency neighborhoods. Decoding reduces to cropping and summing those neighborhoods, which is equivalent to inner products with the carriers.

5.3 Near orthogonal carriers: Gram conditioning and error bounds

When codewords are not exactly orthonormal, define the Gram matrix G with entries G_ij = inner product of h_i and h_j. The matched filter outputs are y = G a (in the noiseless case). The linear minimum variance unbiased estimator for a is
a hat = G inverse times y,
assuming G is invertible. This is the exact left inverse of the encoder restricted to span of the codewords.

Conditioning and robustness

- The sensitivity of a hat to noise is controlled by the condition number kappa(G), the ratio of largest to smallest singular value of G. Smaller kappa means better robustness.
- Mutual coherence mu equals the maximum over i different from j of the absolute value of inner product of h_i and h_j. For unit norm codewords, Gershgorin implies that singular values of G lie in the interval [1 minus (m minus 1) mu, 1 plus (m minus 1) mu] with m equals 6. Thus
  kappa(G) is at most (1 plus 5 mu) divided by (1 minus 5 mu)
  provided 5 mu is less than 1. Designing carriers with small mu gives a well conditioned inverse.
- Noise amplification bound. With additive noise n, the matched outputs become y = G a + b where b_i = inner product of n and h_i. The estimation error satisfies
  norm of (a hat minus a) is at most norm of G inverse times b, which is at most norm of G inverse times norm of b. In an L2 setting, expected squared error per channel is roughly sigma squared times the squared norm of the corresponding row of G inverse.

Design rules of thumb

- Use carriers with disjoint or near disjoint support in the transform domain (tilts, wavelengths) to achieve small coherence.
- If using phase codes, choose masks with low cross correlation and flat spectra.
- Normalize all h_k to unit norm to keep G's diagonal at 1 and to make mu directly meaningful.
- Measure kappa(G) in the implemented system; retune carriers until kappa is close to 1.

Simplex interpretation under near orthogonality

- When working on the normalized 5 simplex plus a global scale, replace the naive inner products by the corrected linear inverse using G inverse, then renormalize to sum to one. This yields stable mixture weights w and a separate scale s even when carriers are only approximately orthogonal.

Summary

- Orthonormal carriers give exact, one step recovery by matched filtering.
- Near orthogonal carriers are fully tractable through the Gram matrix; the only cost is a conditioning factor that can be engineered small.
- All constructions are linear and explicit, so decoding reduces to six inner products plus a 6 by 6 solve, or to FFT crops in the tilt case.

6. Carrier designs and implementations

6.1 Fourier tilt multiplexing with FFT demultiplex

Idea. Assign each class a distinct off-axis plane-wave tilt so that its energy occupies a separate neighborhood in the Fourier domain. The encoder forms a single complex field by summing the six tilted channels. The decoder takes one FFT, crops six disjoint lobes, and integrates to recover the six coordinates.

Practical recipe

1. Pick a complex aperture H by W. Choose six carrier spatial frequencies kx, ky that are well separated, for example a hexagonal pattern on a ring of radius about 0.3 times the Nyquist frequency.

2. Build codewords h_k as unit templates U_k multiplied by exp of i times 2 pi times kx x plus ky y. Normalize to unit energy.
3. Encode E as sum over k of a_k times h_k.
4. Decode: FFT of E, crop a small window around each carrier peak, sum the complex values, divide by the precomputed normalization to get a_k.

Design tips

- Separation. Ensure carrier lobes do not overlap. As a rule of thumb, carrier spacing should exceed the effective bandwidth of the template spectrum times two.
- Windowing. Apply a mild apodization to U_k to reduce spectral leakage.
- Phase stability. If the display or sensor drifts, periodically re-estimate a global phase offset and correct before summation.
- Calibration. Precompute the exact crop masks from a white test field modulated by each tilt. Store their norms for unbiased recovery.
- Dynamic range. If one class is often large, allocate it a slightly larger crop window to reduce truncation error.

Advantages

- One FFT per frame, O(H W log(H W)) time.
- Near orthogonality by design, small Gram off-diagonals.
- Hardware friendly: works on spatial light modulators and standard cameras.

Failure modes and fixes

- Overlap due to undersized ring radius. Increase carrier spacing or shrink template bandwidth by low-pass filtering.
- Wraparound artifacts from circular convolution. Pad the field or center carriers away from FFT boundaries.
- Quantization. Use phase dithering for few-bit modulators.

6.2 Polarization, wavelength, and coded phase variants

Orthogonality can be engineered along physical degrees of freedom other than spatial frequency.

Polarization multiplexing

- Assign six Jones vectors with low mutual coherence, for example three linear angles and three elliptic states that are nearly orthogonal to the first set.
- Encode by summing six fields, each launched with its designated polarization.
- Decode with a rotating analyzer and waveplates or a polarimetric camera to project onto the six analysis states.
- Pros: no spatial bandwidth consumption, works with on-axis holography.
- Cons: practical orthogonality is limited by optics quality; alignment drift introduces Gram off-diagonals.

Wavelength multiplexing

- Assign six center wavelengths if the source and sensors allow, for example RGB plus three near-IR bands.
- Encode by spectral shaping or multi-laser injection.
- Decode with a spectrometer or a filter array.
- Pros: excellent separation, minimal cross-talk.
- Cons: hardware complexity, chromatic dispersion must be calibrated.

Coded phase multiplexing

- Multiply each unit template U_k by a distinct binary or ternary mask with near-flat spectrum and low cross-correlation, such as m-sequences or Legendre sequences lifted to 2D.
- Encode as a simple sum of masked templates, no tilt needed.
- Decode by correlation with the same masks, then scalar projection onto U_k.
- Pros: single optical axis, good resilience to mild defocus, compatible with small fields.
- Cons: correlation cost can be higher than one FFT if implemented naively; choose masks to keep mutual coherence small.

Hybrid designs

- Combine tilt with phase codes to pack more channels at a given ring radius.
- Use polarization to double the channel count with the same tilts.

Implementation checklist

- Normalize every carrier to unit energy so the Gram diagonal is one.
- Measure the Gram matrix experimentally by injecting one carrier at a time; store G inverse for corrected decoding.
- Automate periodic recalibration to track temperature and alignment drift.

6.3 Pure software geometric holography with spherical harmonic signatures

When optics are not required, implement carriers in function space and operate entirely in software. A robust choice is a spherical harmonic signature per class.

Signature construction

1. For each canonical unit template, compute a rotation-invariant descriptor on the unit sphere, for example the magnitude of its spherical harmonic coefficients up to band L. This maps a shape to a vector of coefficients indexed by degree and order.
2. Stack or whiten these coefficients to form U_k.
3. Bind carriers by multiplying U_k elementwise with six orthonormal or low-coherence phase patterns defined over the same coefficient index set. Examples include Hadamard rows, DFT rows, or random Rademacher signs with orthogonalization.

Encoding and decoding

- Encode E as sum over k of a_k times h_k, where h_k is the masked U_k.
- Decode by inner products with each h_k. If Gram is not identity, apply the stored G inverse.
- Recover the six scales, then map each scale back to metric properties using the closed forms in the geometry section.

Design and accuracy notes

- Band limit. Choose L so that the signatures capture the distinctive symmetries. Platonic solids concentrate energy in specific degrees; the sphere is dominated by degree zero.
- Alignment. If input shapes are not already centered and normalized, apply Procrustes alignment or iterative closest rotation in harmonic space to remove pose.
- Stability. Add a small ridge to G before inversion if the smallest singular value is close to zero.

- Speed. Precompute the spherical harmonic transform basis and reuse across shapes. With moderate L, all operations fit comfortably on CPU or GPU.

Advantages

- No hardware constraints, repeatable numerics, straightforward integration into ML pipelines.
- Natural path to mixtures on the 5 simplex plus scale, since the decoding already yields nonnegative weights that can be normalized.

Validation plan

- Unit tests: inject each unit basis vector in the 6D code, verify that decoding returns it to machine precision.
- Noise tests: add Gaussian perturbations to E, measure mean squared error versus predicted Gram-conditioned bounds.
- Cross-family tests: confirm that dual pairs respect known radius and inradius identities after decoding.

In all three modalities, the guiding principle is the same: design carriers so that mutual coherence is small, measure and store Gram for correction, and keep per-channel normalization explicit. This yields fast, stable, and interpretable recovery of the six coordinates in both physical and software settings.

7. Metrics and information geometry on the code

7.1 Euclidean and log Euclidean coordinates

Let a = (a_S, a_T, a_C, a_O, a_D, a_I) with each a_k >= 0.

Euclidean metric

- Distance(a, b) = norm2(a - b).
- Pros: convex, simple, compatible with least squares and PCA.
- Cons: not scale aware. Multiplying all coordinates by a constant changes distances linearly, which may be undesirable when the task cares about ratios.

Log Euclidean metric

- Work on the positive orthant by setting u_k = log a_k for a_k > 0.

- Distance_log(a, b) = norm2(u(a) - u(b)).
- Interpretation: differences are measured in multiplicative factors. A change from 1 to 2 and a change from 10 to 20 have equal length.
- Numerical note: if zeros are possible, either exclude the apex during measurement or use a small epsilon floor a_k >= eps before logging.
- Optimization: many multiplicative models become additive in u, making linear methods appropriate after the log.

Hybrid usage

- Use Euclidean for algorithms that sum coordinates directly (energy, power).
- Use log Euclidean for clustering or regression when relative scale matters more than absolute magnitude.

7.2 Simplex metrics: L1, L2, Fisher

Separate type from size by normalizing a to a nonnegative weight vector w that sums to 1:
s = sum_k a_k, w_k = a_k / s for s > 0.
The weights w live on the 5 simplex Delta5.

Common metrics on Delta5

- L1 (total variation): d1(w, z) = sum_k |w_k - z_k|.
  Interpretation: twice the maximum classification error under optimal coupling. Promotes sparsity and vertex seeking.
- L2 (Euclidean on the simplex): d2(w, z) = sqrt(sum_k (w_k - z_k)^2).
  Interpretation: rotationally symmetric in the affine embedding. Good for smooth interpolation and least squares on probabilities.
- Fisher information metric (Hellinger form):
  Hellinger distance H(w, z) = (1/sqrt(2)) * norm2(sqrt(w) - sqrt(z)).
  Fisher geodesic distance equals 2 * arccos(sum_k sqrt(w_k z_k)).
  Interpretation: natural for probability mixtures; invariant under sufficient statistics mappings and reparametrizations.

Choosing a metric

- If w will be thresholded to one hot, L1 or Hellinger often yields cleaner separation.
- If w will be averaged or combined linearly, L2 is convenient and stable.

- If w is an actual categorical distribution with statistical meaning, use Fisher or Hellinger to respect information geometry.

Practical notes

- Keep s (the global scale) separate. Compare sizes with a scalar metric such as absolute difference in log s or ratio s1/s2.
- For mixed objectives, use a product metric: d_total((w1, s1), (w2, s2)) = d_simplex(w1, w2) plus alpha times |log s1 - log s2|, with alpha tuned to task scale sensitivity.

7.3 Sparsity and one hot selection via L1 penalties

Goal

- Encourage solutions where exactly one class is active at a time (one hot), or at least few classes are active (sparse mixtures), while keeping convexity and interpretability.

L1 regularization on the cone

- Solve: minimize Loss(a) plus lambda times sum_k a_k subject to a_k >= 0.
- Effect: large lambda drives many a_k to zero. This is the standard convex surrogate for the count of nonzeros.
- Scale control: add a simple penalty on s or constrain s to a target range to prevent trivial all zero solutions.

L1 on the simplex

- Work with weights w on Delta5 and an optional separate scale s.
- Solve: minimize Loss(w) plus tau times norm1(w) subject to w in Delta5.
- Since norm1(w) equals 1 on the simplex, replace the penalty by a vertex seeking term such as negative sum_k w_k log w_k (entropy) or a margin based loss that favors a single large coordinate.
- Practical surrogate: minimize Loss(w) minus tau times max_k w_k, which directly encourages a single dominant class while remaining simple to optimize.

Mixed penalties and projected methods

- Group L1 or exclusive Lasso: add gamma times sum_k w_k^2 to discourage sharing mass across coordinates, making one coordinate stand out.

- Projected gradient: alternate a gradient step on w with projection onto Delta5 (nonnegatives that sum to 1).
- For a on the cone, alternate a gradient step with projection onto the nonnegative orthant, and optionally renormalize a to separate w and s.

Stability and calibration

- If the carrier decoding uses a Gram correction, apply the inverse G before thresholding or sparsification so that penalties act on true coordinates, not mixed ones.
- To avoid bias against small but meaningful classes, set lambda or tau adaptively using per channel noise estimates or cross validation.
- When operating in log coordinates, sparsity corresponds to driving u_k to minus infinity; enforce a floor to prevent numeric underflow and then prune entries less than a tuned threshold.

Decision rules

- One hot selection: choose k with largest a_k or largest w_k after decoding and Gram correction.
- Soft selection: keep the top K classes, rescale remaining weights to sum to 1, and carry both K and w forward if mixtures are meaningful.
- Confidence: report the margin w_max minus w_second as a simple confidence score; under Fisher geometry, use the geodesic distance between w and the nearest vertex as an uncertainty measure.

Summary

- Use Euclidean on a when absolute magnitudes matter; use log Euclidean for multiplicative comparisons.
- Use L1, L2, or Fisher on the simplex depending on whether you want separability, smooth averaging, or information theoretic fidelity.
- Promote one hot or sparse mixtures with convex penalties and simple projections, and perform all selection after Gram correction to ensure decisions reflect the true six coordinates.

8. Topology and stratification

8.1 Fan structure and apex singularity

Think of the 6D code space X as the nonnegative orthant in R to the sixth power. It is a convex cone with a single special point at the tip: the origin. This is the apex. The natural stratification of X is by support pattern, meaning which coordinates are zero and which are positive. Each support pattern K subset of {S, T, C, O, D, I} defines a stratum

- X_K = { a in X such that a_k > 0 for k in K and a_j = 0 for j not in K }.
  These strata are relatively open in their coordinate faces and they fit together with the frontier condition: if K is a subset of L then the closure of X_K meets X_L.

Two important fan views:

- Full cone: all 2 to the 6 minus 1 positive faces (every nonempty K) plus the apex. This is a polyhedral fan.
- One hot fan: only the six 1 dimensional cones (the coordinate rays) plus the apex. This is the star of six rays used when exactly one class is allowed at a time.

The apex is a genuine singularity in the stratified sense. Away from it, every point with support K has a neighborhood homeomorphic to a product U times R greater than zero, where U is an open set in the positive orthant of R to the power K. At the apex, any neighborhood is a cone on its link, not a product with R. The link of the apex is the intersection of X with the unit sphere, which is the closed 5 simplex: all nonnegative 6 tuples that sum to 1. This makes the stratification conical: small spheres around the apex see a simplex with faces that mirror the support patterns.

In log coordinates u_k = log a_k for positive a_k, the entire positive part of each stratum is flattened into an affine orthant, and the apex is pushed off to minus infinity. This removes the geometric pinch but leaves the combinatorial stratification intact.

8.2 Components after removing the apex

Connectivity depends on whether you use the full cone or the one hot fan.

- Full cone X minus {0}: connected. Intuitively, once the apex is removed you can move within the same support pattern and also slide between patterns by turning small positive coordinates on or off without ever hitting the origin. The link picture makes this clear: X minus {0} is homeomorphic to (open 5 simplex interior) times R greater than zero together with all of its lower dimensional faces times R greater than zero, which is a connected cone minus its tip.

- One hot fan F minus {0}: six connected components. Each component is an open ray corresponding to a single axis. There are no paths from one axis to another that avoid the apex because any such path would require turning off one coordinate and turning on a different one, which in the one hot model can only happen at the shared point a equals 0.

A useful middle ground is the k sparse model, where at most k coordinates may be nonzero. Removing the apex yields a space with components indexed by unordered k element subsets. For k equal 2 this produces 15 components, each a positive quadrant, which is exactly the setting for the circumsphere curves where the sphere axis and one solid axis are linked.

For algorithms, choose the model that matches the intended behavior:

- pure one hot: decisions are isolated per class;
- sparse: small mixtures are allowed but separated by clear faces;
- full: weights can move freely across the simplex while scale evolves independently.

8.3 Sheaf style organization of invariants over strata

The code organizes naturally as a stratified space, and the shape information you care about is piecewise constant or smoothly varying on these strata. A sheaf style viewpoint lets you write this cleanly and reason about gluing.

Basic assignment

- To each stratum X_K attach the data that is intrinsic to its active classes. Examples:
    - Combinatorics are constant on one hot 1 strata: for T, C, O, D, I record F, E, V and dihedral angles; for S record smooth curvature identities.
    - Metric maps are smooth functions of the active scales: on X_{S,k} carry the closed forms R to a_k and all derived radii, areas, volumes; on higher K carry vector valued functions of the active coordinates.
    - Holographic carriers: attach the set of codewords indexed by K and the local Gram matrix G_K for decoding restricted to span of those carriers.

Restriction and gluing

- If L is a subset of K then restriction from X_K to X_L simply forgets coordinates that go to zero and reduces all formulas accordingly. For example the 2 channel Gram on {S,k} restricts to the 1 channel identity on {k} when the sphere axis is zeroed.

- On open sets that intersect multiple strata, sections are tuples that agree under restriction on overlaps. This encodes the requirement that computations remain consistent as you pass from a 2 sparse region into a 1 sparse region by driving one coordinate to zero.

Constructibility

- The combinatorial part of the sheaf is constructible: it is constant on each stratum and jumps only on lower dimensional faces.
- The metric part is semialgebraic: formulas are polynomial or algebraic in the active coordinates.
- The decoding part is linear-algebraic: inner products and Gram inverses vary smoothly when carriers are fixed.

Why it helps

- Implementation: you can cache per stratum formulas and matrices, and dispatch by support pattern. No branching on pose or type inside the formulas themselves.
- Stability: well defined limits exist as a coordinate tends to zero; restrictions guarantee continuity at the boundary of strata.
- Extensibility: adding new classes or constraints only requires defining their local data and restriction maps. The global behavior then follows from gluing.

In short, the fan gives the right combinatorial skeleton, the apex marks the unique singular point, and a sheaf style organization places the right invariants on each piece with clean restriction rules at the boundaries. This makes both the mathematics and the software crisp, modular, and verifiable.

9. Robustness and numerics

9.1 Noise models and matched filter optimality

Model the observed field as
y = E(a) + n
where E is the linear encoder and n is noise. Useful cases:

- Additive white Gaussian noise
  n is zero mean with covariance sigma squared times identity in the carrier space. With orthonormal carriers, the matched filter outputs are

y_k = inner product of y and h_k = a_k + eta_k
with eta_k independent, zero mean, variance sigma squared. The matched filter is the minimum variance unbiased linear estimator and attains the Cramer Rao bound in this setting.

- Colored additive noise
  Covariance C not equal to identity. Prewhiten with W such that W transpose times W equals C inverse, then apply matched filtering on Wy. Equivalently, use generalized inner products with C inverse.

- Shot noise and read noise in optics
  For photon counting cameras, variance is proportional to intensity plus a read noise floor. Use variance stabilizing transforms such as the Anscombe transform for approximate Gaussianity, or weight inner products by estimated local variance.

- Multiplicative phase or speckle noise
  Model phase drift as a global factor exp of i times theta. Estimate theta by measuring inner product of y with sum of h_k and subtract the phase. For speckle-like multiplicative noise, average multiple frames or apply a short spatial lowpass to the decoded amplitudes.

- Quantization noise
  Treat as uniform noise with variance delta squared over 12 where delta is the quantization step. Dithering before quantization makes this model accurate.

Key takeaways

- Under white additive noise, orthonormal carriers plus simple inner products are optimal among all linear decoders.
- Under nonwhite noise, prewhitening or weighted inner products recover optimality.
- Always normalize carriers to unit energy so the noise variance per channel is the same and directly comparable.

9.2 Mutual coherence design rules

Let G be the Gram matrix with G_ij equal to inner product of h_i and h_j for unit norm carriers.
Define mutual coherence
mu equals the maximum over i not equal to j of absolute value of G_ij.

Why it matters

- Singular values of G lie in the interval from 1 minus 5 mu to 1 plus 5 mu for six channels by Gershgorin.
- Condition number kappa is at most (1 plus 5 mu) divided by (1 minus 5 mu) provided 5 mu is less than 1.
- Noise amplification in the Gram inverse scales with kappa. Keep mu small to control error.

Design rules of thumb

- Tilt carriers
  Place six carrier frequencies on a hexagonal ring well inside the passband. Choose ring radius so that the lobe of each masked template spectrum is at least one lobe width away from its neighbors. Apodize templates to reduce leakage.
- Polarization carriers
  Use near orthogonal Jones vectors. Calibrate the 6 by 6 G from single channel injections and store G inverse for corrected decoding. Recalibrate when temperature or optics change.
- Wavelength carriers
  Ensure spectral filters have minimal overlap at half power. Correct for dispersion by prephase compensation.
- Phase code carriers
  Choose binary or ternary masks with flat spectra and low aperiodic cross correlation, for example m sequences lifted to 2D with orthogonalization. Combine with small tilts to further reduce mu.

Verification and monitoring

- Measure G in the deployed system and compute kappa. Target kappa less than 1.2.
- Track drift by periodically injecting a known a and verifying recovered coordinates after Gram correction.
- If kappa grows, increase carrier separation, reduce template bandwidth, or refresh calibration.

Sparsity interaction

- When you expect one hot states, low mu improves margin between the correct channel and the next best. Under sparse priors, a modest mu still works if the active set is small.

9.3 Quantization, dynamic range, and normalization

Quantization

- Use sufficient bit depth at the field or sensor to make the quantization noise below the desired error. For B bits over a unit range, the effective noise standard deviation is about 2 to the power minus B divided by sqrt(12).
- Add small dithering before quantization to make the noise independent of the signal and to reduce bias.
- For very low bit depth modulators, employ phase dithering or temporal oversampling and average.

Dynamic range

- Prevent clipping in both the encoder and the sensor. If a is expected to vary over orders of magnitude, operate in log coordinates internally and scale the synthesized field so that the largest expected norm fits the device range with headroom.
- Use exposure bracketing or multi gain readouts when available. Combine with linear least squares in the decoded domain.
- In FFT demultiplex, window or pad to avoid wraparound energy that could push peaks into clipping.

Normalization and bias

- Normalize every carrier to unit energy during design so that G has ones on the diagonal.
- Precompute the exact demodulation gain for each channel. In tilt demux, crop windows introduce a gain that depends on window shape; correct by dividing by the measured norm from a unit test hologram.
- In software signatures, whiten the template coefficient vectors so that per degree or per order power is balanced. This reduces sensitivity to bandlimited truncation and improves conditioning.

Scale separation

- After decoding, compute s equals sum of a_k and w equals a divided by s. Apply thresholding and decisions on w, not on a, to avoid conflating type with size. If you need to compare scales across frames or systems, work with log s to make ratios additive and numerically stable.

Bias under nonlinearities

- Gamma curves, saturation, and sensor nonlinearity bias inner products. Calibrate a lookup from digital counts to linear intensity and apply inverse gamma before decoding.
- For phase modulators with nonideal phase response, precompensate the drive with the measured inverse response.

Numerical stability

- When inverting G, add a tiny ridge epsilon on the diagonal if the smallest singular value is close to zero. Choose epsilon far below the noise floor so it acts only as a stabilizer.
- Use 32 bit float for real time decoding and 64 bit float for calibration and G inverse precomputation.
- Cache FFT plans and reuse buffers to avoid allocation jitter in real time systems.

Operational checklist

- Measure G and kappa at install time, store G inverse, and log these values.
- Set bit depth and exposure so that the largest channel uses 60 to 80 percent of full scale.
- Verify per channel gain with single axis injections, then confirm dual pair identities on the circumsphere manifold as ongoing health checks.
- Monitor the margin between the top weight and the runner up on the simplex as a live confidence score.

10. Applications

10.1 Optics: multiplexed CGH for six classes

Use a single complex hologram to carry six independent scale coordinates.

What to build

- Carrier choice: six off axis Fourier tilts on a hex ring, or three tilts times two polarizations.
- Source and modulator: coherent laser, phase SLM or DMD with phase retrieval, camera for capture.
- Templates: precomputed unit fields for sphere and each regular solid, normalized to unit energy.

Pipeline

1. Encode: sum a_k times h_k to form one hologram frame.
2. Display and record: show on the SLM, capture with the camera.
3. Decode: one FFT, crop six peaks, integrate, apply stored gains and the Gram inverse if needed.
4. Map to geometry: convert each recovered a_k into volumes, areas, radii.

Quality controls

- Calibrate the Gram matrix by injecting each carrier alone.
- Use a white field to build exact crop masks and gains.
- Check dual pair identities on circumsphere tests as a health monitor.

Use cases

- Compact storage and transmission of many scenes: a video of holograms becomes a video of six time series.
- Real time identification: live decode and announce which class is present and at what scale.
- Multi user displays: different viewers recover different channels with polarizers or spectral filters.

Metrics to report

- Channel MSE and SNR per frame, condition number of Gram, decode time per frame, cross talk in dB.

10.2 Graphics and vision: rapid which shape detectors

Treat the six axis code as a categorical detector with a scale readout.

Setup

- Build a training set of rendered or scanned shapes aligned to the canonical templates.
- Train a light regressor (small CNN or point cloud network) to output the six a_k directly from images, depth maps, or point sets.
- Postprocess: apply the Gram correction if carriers are used upstream, then separate size and type by s and w.

Runtime

- One forward pass gives a in microseconds to milliseconds.
- Decide one hot by argmax of w, or keep a top K mixture with confidences.

Advantages

- Interpretability: each output dimension has a geometric meaning.
- Speed: six numbers, no large latent space.
- Robustness: angle and texture invariance come for free if inputs are normalized to the templates; residual pose can be handled by a small pose head or prealignment.

Extensions

- Partial views: use silhouette or contour descriptors and match in the six dimensional space.
- Tracking: filter a across time with a Kalman or particle filter to stabilize scale estimates.
- Multi object scenes: segment first, then apply the detector per segment; or sum decoded weights over superpixels.

Benchmarks

- Accuracy of class, MAE of log scale, confusion matrix, and timing on CPU and GPU. Include tests with clutter, occlusion, and noise.

10.3 Machine learning: mixtures on the 5 simplex plus scale

Use the normalized weights on the 5 simplex as a low dimensional, structured latent.

Models

- Classifier with calibrated mixtures: predict w on the simplex and a separate log s. Train with cross entropy on w and L2 on log s.
- Generative decoder: map (w, s) to meshes or images by linearly combining pre learned unit templates and rescaling.
- Semi supervised: few labeled examples define the vertices; unlabeled data cluster near edges and faces.

Regularizers

- Entropy penalty to encourage one hot solutions when appropriate.
- Fisher or Hellinger distance in the loss when comparing predicted and target mixtures.
- Group sparsity to keep mixtures small.

Why it helps

- Stability: the simplex is bounded, gradients stay well behaved.
- Interpretability: interpolation is along edges and faces with clear meaning, not through a black box latent.
- Small data: six parameters can be learned reliably with limited labels.

Tasks

- Shape retrieval: index a database by w and s for sub millisecond k nearest search.
- Few shot adaptation: fine tune only the per class scale maps while keeping the simplex head fixed.
- Programmatic priors: enforce or bias w toward edges that represent valid physical constraints, such as circumsphere pairs.

Reporting

- Calibration curves for w, expected calibration error, simplex geodesic error to nearest vertex, and robustness to domain shift.

10.4 Education: a holographic dictionary of shapes

Turn the six axis code into a hands on lab for geometry.

Interactive module

- Slider for each axis that adjusts scale in real time.
- Toggle for circumsphere mode: setting the sphere radius automatically sets the linked edge length for the chosen solid.
- Live formulas: as a student moves a slider, update area, volume, inradius, circumradius, and dihedral or curvature with exact constants.

Hologram lab

- Display the multiplexed hologram on a classroom SLM.
- Have students place simple analyzers (polarizers or spectral filters) or run a software FFT to demultiplex and read the six numbers.
- Compare dual pairs, verify equal inradii on a common radius, and compute Euler characteristic as a constant per class.

Curriculum hooks

- Similarity and scaling laws: observe area scaling with length squared and volume with length cubed.
- Duality: show cube and octa, dodeca and icosa side by side on the same circumsphere, and measure edge length ratios.
- Probability on the simplex: normalize the six numbers and discuss mixtures versus one hot states.

Assessment

- Short tasks: given a decoded vector, identify the class, compute missing metrics, and explain why combinatorics are unchanged by scale.
- Project: build a miniature dataset of household objects approximating the six classes, regress the six coordinates, and present findings.

Why students like it

- Concrete levers with immediate visual feedback.
- A single consistent representation across optics, software, and math.

- Clear connections from algebraic constants to real measurements.

11. Extensions and comparisons

11.1 Other finite shape alphabets

The 6D code scales directly to any finite catalog.

- Finite catalog with one scale per class
  Replace the six axes by K axes, one per class in the catalog. Examples: Archimedean solids, Johnson solids, prisms and antiprisms, cylinders and cones (with a fixed aspect ratio if you want one scale), curves of constant width (Meissner body), and selected smooth shapes (torus with fixed tube ratio). The space is R nonnegative to the K with the same cone, simplex, and holographic machinery. Carriers become K codewords; decoding remains inner products plus a K by K Gram correction.

- Multi parameter classes
  Some shapes need more than one intrinsic parameter up to similarity, for example a cone with variable aspect ratio or a torus with independent tube ratio. Add one axis per intrinsic parameter of that class, grouped as a block. The global coordinate then has mixed block sizes; a block diagonal decoder still applies because each block feeds only its class. If desired, fix a canonical internal parameter by convention to keep one scale per class and treat other parameters as constrained controls that switch you to a different catalog element.

- Learned alphabets
  When an application does not have a canonical list, learn a finite dictionary of unit templates from data with nonnegative coding and a sparsity prior. After training, freeze the K templates as the alphabet and keep exactly the same linear multiplex and Gram correction. This yields an interpretable, task tuned catalog with the same encoders and decoders.

- Catalog hygiene
  To preserve clean recovery, choose classes whose signatures have low mutual coherence under your chosen carriers. If two classes are nearly indistinguishable under the carrier, either merge them, reparameterize, or add a second carrier modality (for example small tilt plus phase code) to reduce cross talk.

- Constraints beyond circumsphere
  The circumsphere linkage generalizes: tie several classes by any algebraic relation between their scales, for example a common inradius, equal surface area budget, or

fixed volume ratios. Each constraint carves out a low dimensional submanifold in the cone that you can parameterize explicitly and monitor in decoding.

11.2 Adding extrinsic pose as a product factor

Pose lives cleanly outside the intrinsic code.

- Per object pose
  For a rigid object with scale, pose adds 3 translations and 3 rotations. A convenient factorization is
  intrinsic scale axis times SE(3) pose,
  where SE(3) is 3D rigid motions. The sphere's rotation is immaterial, so its pose is 3 translations plus 1 scale. For a regular solid, pose is 3 translation plus 3 rotation plus 1 scale. Keep these factors independent of the intrinsic 6D coordinates.

- Product bundle viewpoint
  The total state for up to K active channels is a stratified bundle over the moduli cone. The base is the intrinsic vector a in R nonnegative to the K. The fiber over a support pattern K active is the product of pose groups for the active classes. Algorithms then operate as: decode a, pick the active set, and estimate poses only for those.

- Shared frame constraints
  Often one wants solids centered on the sphere (circumsphere mode) or a shared world frame. Then translation collapses to one common center, and only rotations (and possibly relative rotations) remain per active solid. With a common circumsphere, the sphere axis gives the radius, translations vanish, and rotations govern only the solids. This reduces extrinsic degrees of freedom and makes estimation better conditioned.

- Estimation pipeline

    1. Decode intrinsic coordinates a and separate w on the simplex and global size s.

    2. Select the active set (one hot or sparse).

    3. Given the chosen class templates, align measured data to the canonical template by Procrustes or an SE(3) solver to estimate translations and rotations.

    4. If circumsphere mode is on, enforce the center constraint and solve only for rotations.
       This separation keeps type and size decisions independent from placement, simplifies uncertainty accounting, and allows reuse of the same pose estimators across classes.

- Counting degrees of freedom
  One active solid: 1 intrinsic scale plus 6 pose equals 7. Sphere alone: 1 plus 3 equals 4. Two active solids centered on a common sphere: 1 shared R plus 3 rotation per solid equals 7 total, independent of world translation. These counts help you design minimal solvers and observability checks.

11.3 Relation to, not reliance on, physics holography

The scheme is a linear code, not a physical conjecture.

- What is shared with the physics metaphor
  Both stories speak of encoding many degrees of freedom into a single object and recovering them by projections. In our case, the single object is a complex field or software signature; the projections are inner products with codewords. There is also an echo of boundary thinking when one uses spherical harmonic signatures, where shape information is summarized on a sphere.

- Where it fundamentally differs
  No area law, no entropy bounds, no gravity, no black hole thermodynamics. There is no claim that information scales with a boundary area or that a lower dimensional theory reproduces higher dimensional physics. The recovery guarantees here come from classical linear algebra: orthonormality implies exact inversion; small mutual coherence implies stable inversion with a Gram correction. Noise performance is set by matched filtering and condition numbers, not by quantum limits.

- Why the analogy can still be useful
  The metaphor helps non specialists remember that a single field can carry multiple independent channels and that simple projections recover them. It also motivates using boundary like signatures (for example, spherical harmonics) when one wants rotation invariance or compact summaries. Beyond that mnemonic value, all claims, proofs, and error bounds rest entirely on constructive coding and standard estimation theory.

- Safe language for publication
  Refer to the construction as multiplexed holography, linear holographic coding, or a holo multiplex. If you mention physics holography at all, do so only as a metaphor and add a line that the present results are independent of any conjectures from quantum gravity.

- Forward looking bridges without overreach
  If desired, you can note technical parallels that remain purely mathematical:

    1. boundary to interior operators like the Dirichlet to Neumann map resemble how spherical signatures determine interior scale parameters;

2. information geometry on the simplex provides a disciplined notion of distance between encoded states.
   These are bridges of method, not claims of physical equivalence.

In summary, the framework extends cleanly to larger or learned catalogs, adds pose as a separate product factor without entangling it with intrinsic type and scale, and uses the holography metaphor only as a helpful narrative. All guarantees come from explicit carrier design, Gram conditioning, and standard matched filtering.

12. Conclusion

12.1 What the six axis code buys in clarity and utility

A single six dimensional nonnegative vector captures the intrinsic geometry of sphere, tetrahedron, cube, octahedron, dodecahedron, and icosahedron without entangling pose, texture, or mesh idiosyncrasies. One number per class is enough, so the representation is small, interpretable, and closed form computable: every metric of interest follows from a scale and fixed constants. The cone structure gives clean selection semantics, the simplex view separates type from size, and the holo multiplex turns six scalars into one field with exact or near exact linear inversion. Dualities are coordinate permutations, circumsphere constraints become one parameter rays, and robustness is governed by familiar quantities like mutual coherence and Gram conditioning. In short, the code is minimal, modular, and measurable: easy to teach, easy to implement, and easy to verify.

12.2 A general template for finite geometric holography

The method is not special to these six classes. For any finite alphabet of shape families with one intrinsic scale each, build a K dimensional nonnegative cone, choose K carriers with low coherence, and use the same encoder and decoder. If some classes need extra intrinsic parameters, assign small blocks per class and keep the carrier map block diagonal. Projectivization always yields a simplex with clear probabilistic semantics; sparsity and selection always reduce to convex penalties and projections. Optical, spectral, polarization, or coded phase carriers are interchangeable with software carriers built from rotation invariant signatures. Across all variants, three rules suffice: normalize carriers, measure the Gram, and correct by its inverse.

12.3 Next steps: open implementation and hardware demo

Open reference stack

- Core library in a small, dependency light package: encode, decode, Gram estimation, simplex utilities, and closed form geometry for all six classes.
- Carriers module with Fourier tilt, coded phase, and spherical harmonic options, each exposing make, calibrate, and demultiplex functions.
- Tests that inject one hot and mixed states, verify exact recovery for orthonormal carriers, and check dual pair identities on the circumsphere manifold.

Reproducible notebooks

- Minimal end to end demos: build carriers, encode a time series of six scalars, decode, plot errors versus predicted Gram bounds, and recover areas, volumes, radii from the decoded scales.
- Education notebook: interactive sliders for the six coordinates, live formulas, and a circumsphere toggle that locks a solid to the sphere radius.

Hardware reference

- Optics bill of materials for a classroom bench: modest laser, phase SLM or DMD, linear polarizers, low cost camera.
- Alignment and calibration scripts: single carrier injections to measure the Gram, FFT crop masks derived from white fields, automatic gain correction.
- Live loop: one hologram per frame, one FFT demultiplex, real time readout of the six scales with SNR and condition number overlays.

Release checklist and metrics

- Git repository with permissive license, unit tests in continuous integration, and exact result hashes for deterministic runs.
- Metrics to report by default: per channel mean squared error, cross talk in dB, Gram condition number, decode latency, and geodesic error on the simplex to the nearest vertex for one hot tasks.

With these pieces in place, the six axis code moves from a clear mathematical construction to a usable tool: a compact language for regular geometry, a reliable multiplex for storage and transmission, and a hands on platform for optics, vision, and education.

References

1. H. S. M. Coxeter, *Regular Polytopes*, 3rd ed., Dover, 1973.
2. Magnus J. Wenninger, *Polyhedron Models*, Cambridge Univ. Press, 1971.
3. Eric W. Weisstein, "Platonic Solid," "Circumradius," "Inradius," MathWorld, Wolfram Research (standard formulas compendium).
4. Joseph W. Goodman, *Introduction to Fourier Optics*, 4th ed., W. H. Freeman, 2017.
5. P. Hariharan, *Basics of Holography*, Cambridge Univ. Press, 2002.
6. Emmett N. Leith and Juris Upatnieks, "Reconstructed wavefronts and communication theory," *JOSA* 52(10), 1962; and "Wavefront reconstruction with continuous-tone objects," *JOSA* 53(12), 1963.
7. B. R. Brown and A. W. Lohmann, "Complex spatial filtering with binary masks," *Applied Optics* 5(6), 1966; and "Computer-generated holograms," *IBM Journal of Research and Development* 13(2), 1969.
8. A. Papoulis, *Systems and Transforms with Applications in Optics*, McGraw-Hill, 1968.
9. Harry L. Van Trees, *Detection, Estimation, and Modulation Theory, Part I*, Wiley, 2001.
10. Steven M. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory*, Prentice Hall, 1993.
11. R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, 2nd ed., Wiley, 2001.
12. Y. C. Eldar and G. Kutyniok (eds.), *Compressed Sensing: Theory and Applications*, Cambridge Univ. Press, 2012.
13. Joel A. Tropp, "Greed is good: algorithmic results for sparse approximation," *IEEE Trans. Information Theory* 50(10), 2004.
14. R. A. Horn and C. R. Johnson, *Matrix Analysis*, 2nd ed., Cambridge Univ. Press, 2012.
15. Shun-ichi Amari, *Information Geometry and Its Applications*, Springer, 2016.
16. J. Lin, "Divergence measures based on the Shannon entropy," *IEEE Trans. Information Theory* 37(1), 1991.
17. K. Kellner, "On Hellinger and Bhattacharyya distances," classic summaries; see also A. Bhattacharyya, "On a measure of divergence between two statistical populations," *Bull. Calcutta Math. Soc.* 35, 1943.

18. P. J. Huber and E. M. Ronchetti, *Robust Statistics*, 2nd ed., Wiley, 2009.

19. D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *IJCV* 60(2), 2004. (general scale-invariance background)

20. R. Osada, T. Funkhouser, B. Chazelle, and D. Dobkin, "Shape distributions," *ACM TOG* 21(4), 2002.

21. M. Kazhdan, T. Funkhouser, and S. Rusinkiewicz, "Rotation invariant spherical harmonic representation of 3D shape," *Symp. Geometry Processing*, 2003.

22. K. Atkinson and W. Han, *Spherical Harmonics and Approximations on the Unit Sphere*, Springer, 2012.

23. J. J. Koenderink and A. J. van Doorn, "Surface shape and curvature scales," *Image and Vision Computing* 10(8), 1992.

24. D. G. Kendall, "Shape manifolds, Procrustean metrics, and complex projective spaces," *Bull. London Math. Soc.* 16(2), 1984.

25. T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham, "Active shape models," *Computer Vision and Image Understanding* 61(1), 1995.

26. J. B. Hestenes and E. Stiefel, classic sources on Procrustes; see also J. C. Gower, "Generalized Procrustes analysis," *Psychometrika* 40(1), 1975.

27. S. W. Golomb and G. Gong, *Signal Design for Good Correlation: For Wireless Communication, Cryptography, and Radar*, Cambridge Univ. Press, 2005. (m-sequences and low-correlation codes)

28. Eugene Hecht, *Optics*, 5th ed., Pearson, 2016. (Jones calculus and polarization analysis)

29. Alan V. Oppenheim and Ronald W. Schafer, *Discrete-Time Signal Processing*, 3rd ed., Pearson, 2010. (FFT practice and windowing)

30. Mark A. Pinsky and Samuel Karlin, *An Introduction to Fourier Series and Integrals*, 2nd ed., AMS, 2002. (Fourier basics relevant to tilt carriers)

Notes:

- Items 1–3 collect exact metric formulas for Platonic solids and are suitable for closing all constants used in Sections 3 and 4.
- Items 4–8, 28–30 cover Fourier optics, CGH, off-axis holography, polarization, and FFT demultiplex implementation details (Sections 5–6).

- Items 9–14 underpin matched filtering, Gram conditioning, Gershgorin bounds, and robustness (Section 9).
- Items 15–18 address information geometry and robust estimation for simplex metrics and noise models (Section 7).
- Items 19–26 provide context for rotation invariance, spherical harmonic signatures, and shape spaces that support the pure software variant (Sections 5–6).
- Item 27 documents low-correlation sequence design for coded phase carriers (Section 6).