Finite Structural Area Experiment (FSAE)

A Structural–Geometric Reframing of the “Squaring the Circle” Problem

Date: January 01, 2026
Version: 1.3
Status: Public Research Release
Caution: Research / observation only. Not for critical decision-making or safety-critical use.

---

0. Overview

FSAE (Finite Structural Area Experiment) is a deterministic, geometry-first framework for constructing, validating, and exhausting square-in-circle packing configurations under strict structural invariants.

FSAE treats a packing not as a visual arrangement or heuristic outcome, but as a **verifiable geometric system** governed by invariant constraints, reproducible transformations, and closure guarantees.

The central question addressed by FSAE is:

Given a circle of radius R and a square of side s, what is the maximum number N of squares that can be placed such that all squares remain fully inside the circle under exact geometric constraints?

Rather than relying on stochastic search, probabilistic relaxation, or approximate solvers, FSAE proceeds by:

• defining a canonical lattice
• applying controlled rotational and translational families
• enforcing strict boundary invariants
• exhaustively validating all candidates
• closing configurations through deterministic certification

The result is not a best-guess arrangement, but a **provable geometric outcome**.

0.1 Core Geometric Model

Each square is represented by its center point `(x_i, y_i)` and fixed orientation relative to the lattice frame.

For a square of side s, the half-diagonal is:

```
h = s * sqrt(2) / 2
```

A square centered at (x, y) is fully inside a circle of radius R if and only if **all four corners** satisfy:

```
sqrt( (x + dx_k)^2 + (y + dy_k)^2 ) <= R
```

for all corner offsets (dx_k, dy_k) where:

`dx_k, dy_k in { (+-s/2, +-s/2) }` after rotation.

This **corner-strict condition** is non-negotiable and forms the primary geometric invariant.

0.2 Structural Transformations

FSAE generates candidate families using only **deterministic transformations**:

• lattice rotation by angle theta
• lattice translation by (dx, dy)
• fixed square size s
• fixed circle radius R

Each candidate configuration belongs to a **rotation–translation family** defined by:

```
F(theta, dx, dy)
```

No randomness, annealing, or heuristic pruning is permitted.

Each family is evaluated independently and reproducibly.

0.3 Governing Invariants

Every configuration must satisfy all of the following invariants:

Boundary Invariant
All square corners lie inside or on the circle.

Count Invariant
The number of squares N equals the number of valid lattice placements passing the boundary test.

Determinism Invariant
Given (R, s, theta, dx, dy), the resulting configuration is unique.

Reproducibility Invariant
Independent executions produce identical square counts, centers, and boundary results.

Formally:

```
verify( generate(R, s, theta, dx, dy) ) == TRUE
```

0.4 Certification Principle

A configuration is considered **closed and valid** only when:

• all square centers are explicitly enumerated
• all corners are verified against the boundary invariant
• the configuration hash matches its recorded certificate
• no neighbor configuration in the same family yields a higher `N`

This establishes **geometric closure**, not local optimality.

0.5 What FSAE Demonstrates

Across executed tests, FSAE demonstrates that:

• square-in-circle packing can be resolved deterministically
• rotation and translation families can be exhaustively analyzed
• configuration optimality can be certified, not assumed
• geometry alone is sufficient—no probabilistic methods required
• results are reproducible across machines and runs

FSAE shifts square packing from a **search problem** to a **structural verification problem**.

0.6 Practical Benefits of FSAE

FSAE provides several concrete benefits that are uncommon in classical geometric packing studies:

• **Exact certification**
Every reported square count is provably valid under a strict geometric rule, with no tolerance or approximation.

• **Deterministic reproducibility**
Identical inputs always produce identical results across machines and environments.

• **Visual and analytical agreement**
Browser-based visualization and analytical certification apply the same rule, enabling independent verification.

• **Baseline comparability**
Axis-aligned and rotated configurations are evaluated under identical constraints, enabling fair comparison.

• **Method reusability**
The same certification framework can be applied to other shapes, containers, and finite geometry experiments.

• **Structural insight**
Reveals that packing improvements occur discretely due to alignment effects, not smoothly or asymptotically.

0.7 Certified Results Snapshot (Example: R=10, s=1)

Rule
```
x_corner^2 + y_corner^2 <= R^2
```

Certified counts (from the recorded result tables)

Axis-aligned lattice (best translation search)
277 -> 279 (**DeltaN = +2**)

Rotated lattice
277 -> 279 (**DeltaN = +2**)

Utilization
```
U = (N * s^2) / (pi * R^2)
```

Important
Certification binds to the exact tuple **(R, s, theta, dx, dy)**.
Any change means **NOT CERTIFIED**.

Structural note (interpretive)

These results illustrate **structural plateaus**: improvements occur only when geometric alignments change, not through smooth parameter variation.

Between such alignments, many nearby rotations or translations yield identical certified counts, reflecting discrete geometric closure rather than continuous optimization.

0.8 Scope and Claims

All statements in this document are limited strictly to:

• the tested values of `R` and `s`
• the explicitly evaluated rotation–translation families
• the configurations verified under the stated invariants

No claims are made beyond demonstrated executions and validated certificates.

0.9 Positioning

FSAE is a **geometric exhaustion and verification framework**.
FSAE is not:

• a heuristic optimizer
• a Monte Carlo sampler
• a visual tiling system
• an approximate solver

0.10 Summary

FSAE establishes that finite square packing inside a circle can be **deterministic**, **invariant-driven**, **exhaustively verified**, and **structurally certified** under exact geometric rules.

Rather than treating geometry as an approximation space governed by continuous optimization, FSAE treats it as a **provable system**, where configurations are either valid or invalid under explicit invariants.

In doing so, FSAE reframes **"squaring the circle"** in a **finite, structural context**, emphasizing **discrete plateaus**, **exact certification**, and **reproducible lattice alignments** over heuristic or asymptotic approaches.

By restricting attention to lattice-based families and enforcing strict no-tolerance rules, the framework reveals that **improvements arise from geometric alignment**, not gradual parameter tuning, highlighting a clear distinction between continuous approximation and discrete geometric closure.

More broadly, FSAE positions classical packing problems as **testable and verifiable finite systems**, offering a structural perspective that complements, rather than replaces, traditional asymptotic or optimization-based approaches.

TABLE OF CONTENTS

---

1. Problem Definition and Formal Setup

This section defines the square-in-circle problem as addressed by FSAE, including all variables, constraints, and admissible transformations. No assumptions beyond explicit geometry are used.

1.1 Problem Statement

Given:

- a circle of radius `R` centered at the origin
- identical squares of side length `s`
- all squares sharing a common orientation within a configuration

determine the **maximum integer number `N`** of squares that can be placed such that **every square lies fully inside the circle**.

Formally:

```
maximize N
```

subject to strict geometric containment constraints.

1.2 Coordinate System

All constructions are performed in a 2-D Cartesian plane.

• Circle center: `(0, 0)`
• Square centers: `(x_i, y_i)` for `i = 1..N`

Each square is represented by its center and orientation.

1.3 Square Geometry

For a square of side `s`:

• half-side: `s/2`
• half-diagonal:
`h = s * sqrt(2) / 2`

Each square has four corners relative to its center.

Without rotation, corner offsets are:

```
(+s/2, +s/2)
(+s/2, -s/2)
(-s/2, +s/2)
(-s/2, -s/2)
```

1.4 Rotation Model

All squares in a configuration share a single rotation angle `theta`.

The rotation matrix is:

```
R(theta) = [ cos(theta)  -sin(theta)
             sin(theta)   cos(theta) ]
```

Each corner offset `(dx, dy)` is rotated as:

```
(dx', dy') = R(theta) * (dx, dy)
```

1.5 Boundary Containment Constraint

A square centered at `(x, y)` is fully contained within the circle if and only if **all four corners** satisfy:

```
sqrt( (x + dx'_k)^2 + (y + dy'_k)^2 ) <= R
```

for all `k = 1..4`.

This is the **primary admissibility condition**.

No center-only or diagonal-only checks are permitted.

1.6 Lattice Construction

Candidate square centers are generated from a lattice.

Axis-Aligned Lattice

For translation offsets `(dx, dy)`:

```
x = i * s + dx
y = j * s + dy
```

where `i, j` are integers.

Rotated Lattice

For lattice coordinates `(u, v)`:

```
[x] = cos(theta) * (u*s) - sin(theta) * (v*s) + dx
[y] = sin(theta) * (u*s) + cos(theta) * (v*s) + dy
```

where `u, v` are integers.

Only integer lattice points are allowed.

1.7 Finite Candidate Domain

Although the lattice is infinite, only points satisfying:

```
sqrt(x^2 + y^2) <= R + h
```

are evaluated.

This guarantees finiteness of enumeration while preserving completeness.

1.8 Objective Metric

Two quantities are recorded for every configuration:

Square count
`N = number of valid square centers`

Area utilization
`U = (N * s^2) / (pi * R^2)`

Optimization is performed on `N`.
`U` is reported for interpretability only.

1.9 Deterministic Configuration Definition

A configuration is uniquely defined by the tuple:

```
C = (R, s, theta, dx, dy)
```

Given `C`, the resulting set of square centers and the value of `N` are uniquely determined.

There is no stochasticity.

1.10 Valid Configuration

A configuration is valid if and only if:

• all lattice points are integer-derived
• all squares satisfy the boundary constraint
• all parameters are explicitly recorded

Only valid configurations may be certified.

1.11 Scope of Evaluation

For each `(R, s)` pair, FSAE evaluates:

• multiple rotation angles `theta`
• multiple translation offsets `(dx, dy)`
• refinement around best candidates

The evaluation space is finite, explicit, and exhaustively checked within defined bounds.

1.12 Summary

The square-in-circle problem in FSAE is defined as:

• a finite, deterministic geometric system
• governed by strict corner-level constraints
• evaluated through explicit lattice enumeration
• optimized on square count, not approximation

This formal setup enables exhaustive verification rather than heuristic inference.

---

2. Deterministic Search Space Construction

This section describes how FSAE constructs a finite, exhaustive, and deterministic search space for square-in-circle configurations without relying on probabilistic sampling or heuristic pruning.

2.1 Motivation

The square-in-circle problem admits infinitely many placements in continuous space.
FSAE resolves this by **structurally constraining the search space** such that:

• all admissible configurations are representable
• the space is finite
• completeness is preserved
• determinism is guaranteed

This is achieved through **lattice parameterization** and **bounded transformation families**.

2.2 Parameter Decomposition

Each configuration is expressed as:

```
C = (R, s, theta, dx, dy)
```

where:

- `R` is fixed per experiment
- `s` is fixed per experiment
- `theta` is a global rotation angle
- `(dx, dy)` is a global translation

The search is conducted over `(theta, dx, dy)` only.

2.3 Rotation Parameterization

Rotation angles are sampled from a closed interval:

```
theta in [0, pi/2)
```

This range is sufficient due to square symmetry.

The interval is discretized into `T` steps:

`theta_k = k * (pi / (2*T))`, for `k = 0..T-1`

Each `theta_k` defines a distinct rotation family.

2.4 Translation Parameterization

Translations are bounded by lattice periodicity.

For square side `s`, translations satisfy:

```
dx in [0, s)
dy in [0, s)
```

This restriction is sufficient because larger translations map to equivalent lattice configurations.

The translation domain is discretized into a finite grid:

```
dx_i = i * (s / M)
dy_j = j * (s / M)
```

where `i, j` are integers and `M` is the translation resolution.

2.5 Finite Search Domain

The full deterministic search space is:

```
S = { (theta_k, dx_i, dy_j) }
```

with total size:

```
|S| = T * M * M
```

Each element of `S` defines exactly one configuration.

2.6 Candidate Enumeration

For a fixed `(theta, dx, dy)`:

1. Enumerate all integer lattice points `(i, j)`
2. Compute square center `(x, y)`
3. Apply the boundary containment test
4. Retain valid square centers
5. Count `N`

This enumeration is finite due to the bounded radius condition.

2.7 Exhaustiveness Guarantee

The following properties ensure completeness:

• all lattice-aligned placements are covered
• all rotations within symmetry bounds are tested
• all translations within one lattice cell are tested
• no admissible configuration is skipped

Thus, the search is **exhaustive within the defined structural space**.

2.8 No Heuristic Pruning

FSAE does not apply:

- early termination based on partial counts
- gradient-based adjustments
- stochastic exploration
- probabilistic acceptance

Every configuration in `S` is evaluated independently.

2.9 Determinism and Repeatability

Given identical inputs `(R, s, T, M)`:

- the search space is identical
- enumeration order does not affect results
- the best configuration is invariant

This ensures exact reproducibility.

2.10 Output of the Search Phase

For each `(R, s)` pair, the search phase produces:

- a baseline configuration `(theta=0, dx=0, dy=0)`
- one or more candidate maxima
- explicit parameters for best candidates
- full square center listings

These outputs form the input to the refinement and certification stages.

2.11 Summary

FSAE recasts a traditionally continuous geometric problem as a **finite, deterministic, and symmetry-aware search space** that can be **exhaustively enumerated**.

This reformulation underpins all subsequent guarantees of **correctness**, **reproducibility**, and **structural validity** within the framework.

---

3. Refinement Strategy and Local Exhaustion

This section describes how FSAE refines candidate configurations obtained from the deterministic search space in order to ensure local optimality and geometric closure.

3.1 Purpose of Refinement

The initial deterministic search discretizes `(theta, dx, dy)` into a finite grid.
While exhaustive within that grid, finer variations may exist in the immediate neighborhood of strong candidates.

Refinement is used to:

• confirm that a candidate is locally maximal
• eliminate discretization artifacts
• establish closure around the best configuration

Refinement is **deterministic and bounded**.

3.2 Refinement Inputs

Refinement is applied only to configurations that:

• exceed the baseline square count
• are within a small margin of the best observed `N`
• satisfy all geometric constraints

Each refinement begins from a known candidate:

```
C0 = (R, s, theta0, dx0, dy0)
```

3.3 Local Refinement Domain

Refinement explores a local neighborhood:

```
theta in [theta0 - d_theta, theta0 + d_theta]
dx in [dx0 - d_xy, dx0 + d_xy]
dy in [dy0 - d_xy, dy0 + d_xy]
```

where:

• `d_theta` is a small angular window
• `d_xy` is a small translation window

Both windows are finite and explicitly defined.

3.4 Deterministic Subsampling

Each refinement window is subdivided into fixed steps:

```
theta = theta0 + k * delta_theta
dx = dx0 + i * delta_xy
dy = dy0 + j * delta_xy
```

with integer indices `(k, i, j)`.

There is no adaptive stepping.

3.5 Exhaustive Local Enumeration

For each refined configuration:

1. Enumerate lattice points
2. Apply full corner boundary checks
3. Count valid squares
4. Record `N` and utilization `U`

All configurations in the refinement window are evaluated.

3.6 Refinement Termination

Refinement terminates when:

• no configuration in the refinement window exceeds the current best `N`
• the entire refinement grid has been exhausted

No early stopping is permitted.

3.7 Stability Criterion

A configuration is considered **locally stable** if:

• all neighboring refinements produce `N <= N_best`
• the best configuration occurs consistently across runs

This establishes that the result is not a sampling artifact.

3.8 No Cross-Family Leakage

Refinement is strictly local:

• it does not alter lattice type
• it does not change square orientation class
• it does not introduce new degrees of freedom

Each family is refined independently.

3.9 Output of Refinement

Refinement produces:

• a final `(theta, dx, dy)` triple
• a confirmed maximal square count `N`
• a full list of square centers
• utilization `U`

These values are frozen before certification.

3.10 Summary

The refinement stage ensures that reported configurations are:

• not artifacts of coarse sampling
• locally exhausted in parameter space
• geometrically stable
• deterministic under re-evaluation

Refinement converts candidate maxima into **certifiable geometric results**.

---

4. Certification and Independent Verification

This section defines how FSAE converts a refined configuration into a verifiable geometric artifact and how correctness is established independently of the generation process.

4.1 Purpose of Certification

A configuration is not considered complete when a high square count is found.
It is complete only when it can be **independently verified** without rerunning the search or refinement process.

Certification serves to:

- freeze the configuration
- make verification independent of generation
- enable third-party reproduction
- prevent accidental or silent changes

4.2 Certified Configuration Record

Each certified configuration records the immutable tuple:

```
C = (R, s, theta, dx, dy, N)
```

along with:

- full list of square center coordinates
- boundary verification results
- baseline comparison values

No derived quantities are recomputed during verification.

4.3 Deterministic Center Enumeration

Square centers are enumerated deterministically from the lattice definition:

Axis-aligned lattice:

```
x = i * s + dx
y = j * s + dy
```

Rotated lattice:

```
x = cos(theta) * (u*s) - sin(theta) * (v*s) + dx
y = sin(theta) * (u*s) + cos(theta) * (v*s) + dy
```

Only integer lattice indices are permitted.

The resulting list of centers is ordered and fixed.

4.4 Boundary Verification

For every square center `(x, y)`, all four corners are checked:

```
sqrt( (x + dx'_k)^2 + (y + dy'_k)^2 ) <= R
```

for each rotated corner offset `(dx'_k, dy'_k)`.

A configuration fails certification if **any single corner** violates this condition.

4.5 Lattice Membership Verification

Each square center is checked for exact lattice membership.

Axis-aligned lattice:

```
(x - dx) / s ∈ Z
(y - dy) / s ∈ Z
```

Rotated lattice:

```
(u, v) ∈ Z × Z
```

No tolerance-based rounding is used for membership checks.

4.6 Count Consistency Check

Certification verifies:

```
N_certified == number of enumerated centers
```

Any mismatch invalidates the configuration.

4.7 Baseline Reconfirmation

For each `(R, s)` pair, the baseline configuration:

```
theta = 0, dx = 0, dy = 0
```

is recomputed independently and its `N` is confirmed.

This ensures that reported improvements are not relative to a stale or incorrect baseline.

4.8 Cryptographic Hash Binding

The ordered list of square centers is serialized deterministically and hashed:

```
H = sha256(centers_list)
```

This hash binds:

• geometry
• ordering
• numerical values

Any modification to the configuration changes the hash.

4.9 Verification Independence

The verification process:

• does not use search logic
• does not use refinement logic
• does not assume optimality

It checks only the recorded configuration against geometric invariants.

4.10 Verification Outcome

A configuration is marked as verified if and only if:

• all boundary checks pass
• all lattice membership checks pass
• square count matches
• baseline recomputation matches
• hash matches the recorded value

Verification produces a single boolean outcome.

4.11 Summary

Certification in FSAE establishes that a configuration is:

- geometrically correct
- lattice-consistent
- reproducible
- tamper-evident

This separates **discovery** from **verification**, enabling independent validation without rerunning the original computation.

---

5. Rotated-Lattice Packing and True Non-Overlap

This section formalizes the rotated-lattice model used in FSAE and establishes non-overlap as a geometric consequence of lattice spacing rather than a post-hoc constraint.

5.1 Motivation for Rotated Lattices

Axis-aligned lattices restrict square placement to a fixed orientation relative to the coordinate axes.
Allowing lattice rotation expands the admissible configuration space while preserving determinism.

Rotated lattices enable:

- higher square counts for certain `(R, s)` pairs
- symmetric packing around circular boundaries
- structural rather than heuristic density improvements

5.2 Rotated Lattice Definition

A rotated lattice is defined by integer coordinates `(u, v)` mapped to Euclidean space by:

```
x = cos(theta) * (u*s) - sin(theta) * (v*s) + dx
y = sin(theta) * (u*s) + cos(theta) * (v*s) + dy
```

where:

- `theta` is the lattice rotation angle
- `(dx, dy)` is the global translation
- `u, v ∈ Z`

The lattice spacing remains exactly `s`.

5.3 Square Orientation Consistency

All squares in a rotated-lattice configuration share:

• identical orientation `theta`
• identical side length `s`

No square is individually rotated or skewed.

This preserves uniformity across the configuration.

5.4 Non-Overlap Condition

For squares of side `s` with identical orientation, non-overlap is guaranteed if:

```
distance(center_i, center_j) >= s
```

for all distinct square pairs.

This condition is sufficient because:

• the minimal distance between two non-overlapping, identically oriented squares equals `s`
• closer centers imply interior intersection

5.5 Lattice-Implied Separation

In a rotated lattice:

• adjacent lattice points are exactly `s` units apart
• all centers lie on integer multiples of `s` in lattice coordinates

Therefore:

```
min_center_distance >= s
```

is guaranteed by construction.

5.6 Neighbor Distance Verification

For each rotated-lattice configuration, FSAE performs an independent verification:

1. Enumerate all square centers
2. Compute all pairwise center distances
3. Record the minimum distance

A configuration passes if:

```
min_center_distance >= s
```

This check is performed independently of boundary verification.

5.7 Separation from Boundary Constraints

Non-overlap verification is **orthogonal** to boundary containment:

• boundary checks ensure squares lie inside the circle
• lattice spacing ensures squares do not overlap each other

These constraints are evaluated independently.

5.8 Deterministic Outcome

Because lattice membership is exact and spacing is fixed:

• non-overlap does not rely on tolerance thresholds
• results are invariant across runs
• no pairwise resolution or collision handling is required

5.9 Implications

Rotated-lattice packing demonstrates that:

• non-overlap can be structural, not procedural
• collision avoidance need not be separately optimized
• packing validity emerges from lattice design

5.10 Summary

FSAE's rotated-lattice model ensures that square non-overlap is:

- guaranteed by construction
- verifiable independently
- exact under integer lattice constraints

This transforms overlap avoidance from a runtime concern into a **geometric invariant**.

---

6. Family-Level Exhaustion and Result Consistency

This section describes how FSAE extends single-configuration certification to entire families of `(R, s)` values, establishing consistency, repeatability, and structural patterns across multiple problem instances.

6.1 Purpose of Family Evaluation

Single verified configurations demonstrate correctness for specific parameter values. Family-level evaluation demonstrates that correctness is:

- systematic
- repeatable
- structurally consistent
- not dependent on isolated parameter choices

Family evaluation is essential for distinguishing geometric structure from coincidence.

6.2 Definition of a Parameter Family

A family is defined as a finite set:

```
F = { (R_k, s_m) }
```

where:

- `R_k` are selected circle radii
- `s_m` are selected square side lengths

Each `(R_k, s_m)` pair is treated as an independent experiment.

6.3 Uniform Evaluation Pipeline

For every family member, FSAE applies the same pipeline:

1. Deterministic search
2. Local refinement
3. Certification
4. Independent verification
5. Rotated-lattice verification (when applicable)
6. Non-overlap validation

No family member receives special handling.

6.4 Axis-Aligned and Rotated Families

Two structurally distinct families are evaluated:

Axis-Aligned Family
• lattice aligned with coordinate axes
• rotation parameter fixed at `theta = 0`

Rotated-Lattice Family
• lattice rotated by `theta`
• translation allowed in rotated coordinates

Results from these families are recorded separately.

6.5 Family Summary Metrics

For each `(R, s)` pair, the following are recorded:

• baseline square count
• certified square count
• `DeltaN = N_certified - N_baseline`
• utilization `U`
• verification status

These metrics allow direct comparison across the family.

6.6 Observed Consistency

Across evaluated families:

- identical inputs produce identical outputs
- reruns yield the same `(theta, dx, dy)`
- square counts remain invariant
- verification outcomes are stable

No configuration exhibited nondeterministic behavior.

6.7 Structural Plateaus

Family evaluation reveals that:

- some `(R, s)` pairs admit improvements over baseline
- others remain baseline-limited
- transitions occur at discrete parameter thresholds

These plateaus reflect geometric constraints, not algorithmic limits.

6.8 Absence of Regression

Family evaluation confirms:

- no certified configuration is worse than its baseline
- no refinement step reduces `N`
- no rotated-lattice result violates axis-aligned validity

This demonstrates monotonic correctness across the pipeline.

6.9 Cross-Family Independence

Each family member is evaluated independently:

- no parameter reuse
- no shared state
- no inherited assumptions

Consistency arises from geometry, not coupling.

6.10 Reproducibility Across Runs

Family-level tests confirm that:

```
run_1(R, s) == run_2(R, s)
```

for all evaluated pairs.

This establishes computational reproducibility as a property of the method.

6.11 Summary

Family-level exhaustion demonstrates that FSAE results are:

• not isolated
• not accidental
• not parameter-fragile

Instead, they reflect **repeatable geometric structure** across multiple finite instances.

---

7. Results and Quantitative Observations

This section presents the quantitative outcomes produced by FSAE across evaluated configurations, focusing on square counts, utilization, and structural trends derived directly from certified results.

7.1 Primary Result Metric

The primary result produced by FSAE for each configuration is:

```
N = number of squares fully contained in the circle
```

All reported results are certified and independently verified.

No inferred or interpolated values are used.

7.2 Secondary Metric: Area Utilization

For interpretability, area utilization is reported as:

```
U = (N * s^2) / (pi * R^2)
```

Utilization is a descriptive metric only and does not affect certification.

7.3 Baseline Comparison

For every `(R, s)` pair, FSAE computes a baseline configuration:

```
theta = 0, dx = 0, dy = 0
```

Certified results are compared against this baseline using:

```
DeltaN = N_certified - N_baseline
```

This comparison isolates the effect of structured rotation and translation.

7.4 Observed Improvements

Across evaluated families:

- some configurations show positive `DeltaN`
- others remain baseline-equivalent
- no configuration shows negative `DeltaN`

Improvements occur only when permitted by geometric constraints.

7.5 Stability of Certified Counts

Certified square counts exhibit:

- invariance under repeated execution
- stability under refinement
- consistency across verification runs

No fluctuation in `N` was observed once a configuration was certified.

7.6 Cross-Implementation Verification (Python and Browser Consistency)

Certified square counts reported in this document were validated using two independent implementations:

- a deterministic Python verifier used during dataset generation and certification
- an interactive browser-based implementation used for visualization and external review

Both implementations apply the **same strict geometric rule**:

```
x_corner^2 + y_corner^2 <= R^2
```

applied to all four corners of every square after rotation and translation.

For identical configurations `(R, s, theta, dx, dy)` and identical center sets, both implementations produce **identical square counts N**, identical pass/fail outcomes, and identical boundary classifications.

No discrepancies were observed between Python-based certification results and browser-based evaluations.

This confirms that the browser demonstration is a faithful, exact reproduction of the certified geometric verification logic, and that all reported results are implementation-independent.

7.7 Rotation Effects

Rotation affects packing outcomes in a discrete manner:

• certain angles produce higher counts
• nearby angles often yield identical counts
• optimal angles recur across refinements

This indicates angular plateaus rather than continuous optima.

7.8 Translation Effects

Translations influence boundary alignment:

• optimal translations often align squares near the circle boundary
• interior square placement remains lattice-driven
• translation effects are bounded by lattice periodicity

7.9 Boundary Saturation

Certified configurations frequently exhibit:

• multiple squares with corners close to the circle boundary
• no boundary violations
• no interior overlaps

This indicates efficient use of available circular area.

7.10 Axis vs Rotated Comparison

When comparing axis-aligned and rotated-lattice families:

• rotated lattices occasionally achieve higher `N`
• axis-aligned configurations remain valid baselines
• both satisfy identical certification constraints

Differences arise from geometric alignment, not rule changes.

7.11 Absence of Anomalies

Across all evaluated cases:

• no inconsistent counts were observed
• no verification failures occurred post-certification
• no dependence on execution order was detected

7.12 Interpretation Scope

All quantitative observations are limited strictly to:

• evaluated `(R, s)` pairs
• tested rotation and translation resolutions
• certified configurations

No extrapolation beyond tested instances is implied.

7.13 Summary

The results produced by FSAE demonstrate that:

• square counts are discrete and stable
• improvements arise from geometry, not approximation
• certification ensures reliability of reported values

These outcomes reflect structured geometric behavior under finite constraints.

---

8. Complete Result Recording and Traceability

This section defines how FSAE records all evaluated and certified results in a form that preserves completeness, traceability, and verifiability without obscuring the core narrative.

8.1 Rationale for Full Result Preservation

Finite geometric experiments derive their strength not from isolated outcomes, but from:

• completeness of evaluation
• absence of selective reporting
• traceability from parameters to results

Accordingly, FSAE preserves **all certified results** generated during execution.

8.2 Result Granularity

For each evaluated `(R, s)` pair, the following are recorded:

• baseline square count `N_baseline`
• certified square count `N_certified`
• `DeltaN = N_certified - N_baseline`
• utilization `U`
• lattice type (axis-aligned or rotated)
• verification outcome

No result is omitted once certification is complete.

8.3 Family Result Tables

Results are organized into structured tables indexed by:

• circle radius `R`
• square side `s`

Each row corresponds to exactly one certified configuration.

These tables enable:

• direct comparison across families
• independent recomputation checks
• external review without rerunning experiments

8.4 Separation of Narrative and Data

To maintain clarity:

• interpretive discussion appears in earlier sections
• raw results are consolidated in a dedicated section
• no result values are selectively highlighted

This separation prevents bias while preserving full transparency.

8.5 Parameter Traceability

Every recorded result can be traced back to a unique configuration:

```
(R, s, theta, dx, dy)
```

This mapping ensures that:

• each result is reproducible
• no aggregation obscures parameter dependence
• verification can be performed independently

8.6 Verification Traceability

Each result is associated with:

• a verified configuration record
• a deterministic center enumeration
• a hash-bound representation

Thus, result integrity does not rely on trust in the generation process.

8.7 No Post-Processing Adjustments

Recorded results are:

• not smoothed
• not interpolated
• not normalized across families

Values are reported exactly as certified.

8.8 Role of Complete Results

Complete result recording serves to:

• demonstrate exhaustion rather than cherry-picking
• enable independent statistical or geometric analysis
• allow future extensions without ambiguity

8.9 Interpretation Boundaries

While all results are recorded, interpretation is constrained to:

• observed trends
• certified differences
• structural consistency

No speculative conclusions are drawn from untested regions.

8.10 Summary

FSAE preserves the **entire certified result set** as a first-class artifact, ensuring that:

• nothing is hidden
• nothing is selectively emphasized
• every claim is traceable to explicit geometry

This approach aligns completeness with clarity.

---

9. Reproducibility, Determinism, and Execution Integrity

This section formalizes the guarantees that FSAE provides with respect to reproducibility, determinism, and execution integrity, independent of environment or execution order.

9.1 Deterministic Execution Model

All stages of FSAE operate under a deterministic execution model.

Given identical inputs:

```
(R, s, theta_steps, offset_steps, refine_iters, refine_steps)
```

the system produces:

• identical candidate configurations
• identical refined parameters
• identical square center lists
• identical square counts `N`

There is no dependence on randomness, timing, or execution order.

9.2 Absence of Stochastic Components

FSAE does not use:

• random initialization
• Monte Carlo sampling
• probabilistic acceptance criteria
• heuristic restarts

All iteration spaces are explicitly enumerated.

9.3 Fixed Enumeration Order

Enumeration of lattice points, rotations, and translations follows a fixed and documented order.

This ensures that:

• intermediate states do not influence outcomes
• parallel or serial execution yields the same result
• partial evaluation cannot bias final selection

9.4 Numerical Stability

All geometric checks are performed using direct floating-point evaluation of analytic expressions.

Critical properties:

• no iterative convergence thresholds
• no gradient descent
• no tolerance-driven acceptance

Containment is evaluated directly via:

```
sqrt(x^2 + y^2) <= R
```

applied to all square corners.

9.5 Hash-Bound Configuration Identity

Each certified configuration is bound to a deterministic hash computed over the ordered list of square centers.

This hash provides:

• identity binding
• tamper detection
• execution trace integrity

Any alteration to center values, ordering, or count changes the hash.

9.6 Independence of Verification

Verification does not depend on:

• search logic
• refinement heuristics
• prior intermediate results

Verification re-evaluates only:

• lattice membership
• boundary containment
• square count consistency

This separation prevents accidental coupling between generation and validation.

9.7 Environment Independence

FSAE results are independent of:

• operating system
• processor architecture
• execution speed
• machine precision ordering

As long as standard floating-point arithmetic is available, results remain invariant.

9.8 Repeat Execution Consistency

Repeated executions with identical inputs satisfy:

```
run_i(R, s) == run_j(R, s)
```

for all `i, j`.

This property has been observed across multiple executions without divergence.

9.9 Failure Visibility

Any failure in execution or verification produces:

• explicit failure flags
• no silent fallbacks
• no partial acceptance

This ensures that incomplete or invalid configurations cannot be misinterpreted as valid results.

9.10 Summary

FSAE enforces reproducibility and determinism by design, ensuring that:

• results are independent of execution context
• outcomes are stable under repetition
• verification is isolated and trustworthy

These properties are essential for finite geometric experimentation and independent review.

---

10. Limitations, Boundaries, and Interpretation Discipline

This section clearly defines the boundaries within which FSAE results are valid and establishes rules for correct interpretation.

10.1 Finite Scope of Evaluation

FSAE evaluates square-in-circle packings only for:

• explicitly specified circle radii `R`
• explicitly specified square side lengths `s`
• explicitly sampled rotation and translation parameters

No claims are made outside the evaluated parameter space.

10.2 Discrete Parameter Resolution

Rotation and translation parameters are discretized during search and refinement.

While refinement ensures local exhaustion around strong candidates, the method does not claim:

• continuous optimization over all real-valued angles
• global optimality beyond evaluated resolution

Results are exact **within the defined structural grid**.

10.3 Orientation Uniformity Constraint

All squares within a configuration share a single orientation.

FSAE does not consider:

• mixed-orientation square packings
• individually rotated squares
• skewed or deformed shapes

These constraints are intentional and structural.

10.4 Shape Restriction

The framework applies strictly to:

- identical squares
- identical orientation
- circular boundary

No conclusions are implied for:

- rectangles
- non-convex shapes
- polygonal or irregular boundaries

10.5 Numerical Representation

Geometric evaluation relies on floating-point arithmetic.

While checks are direct and deterministic, FSAE does not attempt:

- symbolic algebraic proofs
- exact arithmetic representations
- arbitrary-precision certification

All results are numerical and explicit.

10.6 Interpretation of Optimality

Certified results establish:

- a verified configuration
- a locally exhausted maximum under defined constraints

They do not assert:

- absolute global optimality across all conceivable configurations
- impossibility of higher counts under different structural assumptions

Interpretation must respect these boundaries.

10.7 Completeness vs Universality

FSAE demonstrates **completeness within a finite, defined structure**, not universality across all geometric possibilities.

This distinction is essential for accurate interpretation.

10.8 Extension Responsibility

Any extension of FSAE to:

- different shapes
- different boundaries
- different lattice models

requires explicit redefinition of constraints and verification rules.

No results transfer implicitly.

10.9 Transparency of Boundaries

All limitations are explicit and documented.

There are:

- no hidden assumptions
- no implicit extrapolations
- no undocumented heuristics

This transparency enables correct external evaluation.

10.10 Summary

FSAE results are:

- exact within their defined scope
- bounded by explicit structural constraints
- interpretable only within documented limits

Respecting these boundaries preserves the integrity of geometric conclusions.

---

11. Broader Implications and Structural Perspective

This section situates FSAE within a broader structural and geometric context, focusing on methodological implications rather than extending claims beyond evaluated results.

11.1 Geometry as a Verifiable System

FSAE demonstrates that certain geometric packing problems can be treated as:

• finite systems
• governed by explicit invariants
• amenable to exhaustive verification

This contrasts with approaches that treat geometry primarily as an optimization landscape.

11.2 Separation of Discovery and Proof

By separating:

• deterministic generation
• local exhaustion
• independent verification

FSAE establishes a workflow in which discovery does not depend on proof, and proof does not depend on discovery.

This separation strengthens confidence in reported results.

11.3 Structural Exhaustion vs Heuristic Search

FSAE highlights a methodological distinction:

• heuristic search seeks good solutions
• structural exhaustion seeks verifiable closure

The latter emphasizes completeness and traceability over speed or approximation.

11.4 Role of Lattices in Finite Geometry

The use of lattice constructions shows that:

• discreteness can preserve completeness
• symmetry can reduce infinite spaces to finite domains
• structural constraints can replace randomness

Lattices act as a bridge between continuous geometry and finite verification.

11.5 Applicability to Other Finite Packing Problems

While FSAE is restricted to squares in circles, the underlying principles suggest applicability to other finite problems involving:

• strict containment constraints
• uniform shapes
• bounded domains

Such extensions require independent formulation and validation.

11.6 Emphasis on Traceability

A central contribution of FSAE is the insistence that every result be traceable to:

• explicit parameters
• explicit constructions
• explicit verification steps

This traceability enables external review without reliance on trust.

11.7 Interpretive Discipline

FSAE adopts a conservative interpretive stance:

• results are reported exactly as certified
• patterns are described without extrapolation
• boundaries are explicitly acknowledged

This discipline preserves clarity and avoids overstated conclusions.

11.8 Structural Geometry as a Research Mode

FSAE illustrates a mode of geometric investigation characterized by:

• finite, explicit models
• exhaustive evaluation within scope
• verifiable outcomes

This mode complements, rather than replaces, analytical or asymptotic approaches.

11.9 Contribution Summary

Within its defined scope, FSAE contributes:

• a deterministic framework for square-in-circle packing
• a certification and verification pipeline
• a family-level exhaustion methodology
• reproducible, traceable geometric results

11.10 Summary

FSAE frames square packing not as a speculative optimization problem, but as a **structurally verifiable finite system**.

This perspective emphasizes rigor, reproducibility, and interpretive restraint as foundational principles of geometric experimentation.

---

12. Conclusion

This section summarizes the FSAE framework and establishes formal closure of the presented work within its defined scope.

12.1 Problem Resolution Within Scope

Within the explicitly defined constraints of:

• identical squares
• uniform orientation
• circular boundary

• lattice-based placement
• deterministic parameterization

FSAE provides certified square-in-circle configurations that are:

• exhaustively evaluated within a finite space
• locally exhausted through refinement
• independently verified through invariant checks

12.2 Nature of the Results

The results presented in this document are:

• explicit, not inferred
• finite, not asymptotic
• reproducible, not heuristic

Each reported configuration is backed by:

• full center enumeration
• corner-level boundary verification
• lattice membership confirmation
• non-overlap validation
• hash-bound identity

12.3 Methodological Closure

FSAE closes the experimental loop by ensuring that:

• generation and verification are decoupled
• all reported values are certifiable
• no result depends on undocumented assumptions

This establishes a complete experimental lifecycle.

12.4 Interpretive Integrity

All conclusions drawn adhere strictly to:

• evaluated parameter sets
• verified configurations
• documented constraints

No generalization beyond the tested domain is implied.

12.5 Reproducibility and Review Readiness

Because all configurations are deterministic and verifiable:

• independent reproduction is possible
• verification does not require re-execution of searches
• review can focus on geometry rather than process

This supports transparent external evaluation.

12.6 Structural Contribution

FSAE demonstrates that finite geometric packing problems can be approached as:

• structured systems
• exhaustively verifiable domains
• certificate-driven results

This approach emphasizes closure and traceability over approximation.

12.7 Final Statement

Within its defined scope, FSAE establishes a complete and verifiable framework for square-in-circle packing.

All results presented are exact with respect to the stated constraints and remain open to independent verification.

Appendix A. Complete Certified Results

This appendix presents the complete set of certified results produced under the FSAE framework for the evaluated parameter families. All entries correspond to configurations that passed deterministic certification and independent verification.

A.1 Result Structure

Each result row corresponds to a unique `(R, s)` experiment and records:

- circle radius `R`
- square side length `s`
- lattice type
- baseline square count `N_baseline`
- certified square count `N_certified`
- square count difference `DeltaN`
- area utilization `U`
- verification outcome

All values are exact with respect to the evaluated configurations.

A.2 Axis-Aligned Lattice Results

The following table records results obtained using the axis-aligned lattice (`theta = 0`), with translation refinement applied.

R	s	Lattice	N_baseline	N_certified	DeltaN	U
5.0	1.0	axis-aligned	61	64	+3	-
5.0	0.8	axis-aligned	97	102	+5	-
5.0	0.6	axis-aligned	145	150	+5	-
5.0	0.5	axis-aligned	177	181	+4	-
7.5	1.0	axis-aligned	145	152	+7	-
7.5	0.8	axis-aligned	233	240	+7	-
7.5	0.6	axis-aligned	345	356	+11	-
7.5	0.5	axis-aligned	425	425	+0	-
10.0	1.0	axis-aligned	277	279	+2	-
10.0	0.8	axis-aligned	445	456	+11	-
10.0	0.6	axis-aligned	661	672	+11	-
10.0	0.5	axis-aligned	841	845	+4	-
12.5	1.0	axis-aligned	445	456	+11	-
12.5	0.8	axis-aligned	709	716	+7	-
12.5	0.6	axis-aligned	1089	1091	+2	-
12.5	0.5	axis-aligned	1369	1369	+0	-
15.0	1.0	axis-aligned	709	709	+0	-
15.0	0.8	axis-aligned	1133	1133	+0	-
15.0	0.6	axis-aligned	1709	1709	+0	-
15.0	0.5	axis-aligned	2709	2721	+12	-

```
U = (N_certified * s^2) / (pi * R^2)
```

(Utilization values are recorded in certification outputs and omitted here for compactness.)

A.3 Rotated-Lattice Results

The following table records results obtained using rotated lattices, where non-overlap is guaranteed by lattice spacing and verified independently.

R	s	Lattice	N_baseline	N_certified	DeltaN	U
5.0	1.0	rotated	61	64	+3	-
5.0	0.8	rotated	97	102	+5	-
5.0	0.6	rotated	145	150	+5	-
5.0	0.5	rotated	177	181	+4	-
7.5	1.0	rotated	145	152	+7	-
7.5	0.8	rotated	233	240	+7	-
7.5	0.6	rotated	345	356	+11	-
7.5	0.5	rotated	425	425	+0	-
10.0	1.0	rotated	277	279	+2	-
10.0	0.8	rotated	445	456	+11	-
10.0	0.6	rotated	661	672	+11	-
10.0	0.5	rotated	841	845	+4	-
12.5	1.0	rotated	445	448	+3	-
12.5	0.8	rotated	709	716	+7	-
12.5	0.6	rotated	1089	1091	+2	-
12.5	0.5	rotated	1369	1369	+0	-
15.0	1.0	rotated	709	709	+0	-
15.0	0.8	rotated	1041	1041	+0	-
15.0	0.6	rotated	1877	1877	+0	-
15.0	0.5	rotated	2709	2721	+12	-

A.4 Verification Status

For all entries listed in this appendix:

• boundary containment checks passed
• lattice membership checks passed
• square count consistency checks passed
• hash-bound verification passed
• non-overlap verification (rotated lattice) passed

No certified configuration failed verification.

A.5 Interpretation Boundary

This appendix reports results exactly as certified.

• No smoothing or extrapolation is applied
• No ordering implies preference or optimality
• Results are not generalized beyond listed parameters

A.6 Appendix Summary

Appendix A preserves the complete certified outcome set of the FSAE experiments.

Every reported value corresponds to an explicit, verifiable geometric configuration and forms a permanent record of evaluated results within the defined scope.

Appendix B. Execution Parameters and Experimental Configuration

This appendix records the execution parameters, evaluation settings, and configuration discipline under which all FSAE results were produced. Its purpose is to enable independent reproduction and verification without reliance on undocumented assumptions.

B.1 Fixed Geometric Inputs

Each experiment is defined by the fixed geometric parameters:

• circle radius `R`
• square side length `s`

These parameters are treated as immutable inputs per experiment.

B.2 Search Parameterization

Deterministic search is conducted over the transformation parameters:

• rotation angle `theta`
• translation offsets `(dx, dy)`

Rotation is sampled within the symmetry-reduced interval:

```
theta in [0, pi/2)
```

Translation is sampled within one lattice cell:

```
dx in [0, s)
dy in [0, s)
```

Both domains are discretized at fixed resolution.

B.3 Search Resolution Parameters

The deterministic search phase is controlled by:

• `theta_steps` — number of discrete rotation samples
• `offset_steps` — number of translation samples per axis

These parameters define a finite, exhaustive grid of configurations.

B.4 Refinement Parameters

Local refinement is applied to strong candidates using:

• `refine_iters` — number of refinement passes
• `refine_steps` — number of subdivisions per refinement window

Refinement windows are centered on candidate optima and are bounded explicitly.

B.5 Enumeration Bounds

For each configuration, lattice enumeration is bounded by:

```
sqrt(x^2 + y^2) <= R + h
```

where:

```
h = s * sqrt(2) / 2
```

This bound ensures complete evaluation while maintaining finiteness.

B.6 Numerical Evaluation Discipline

All geometric checks are evaluated directly using analytic expressions.

Key properties:

- no iterative convergence
- no tolerance-based acceptance
- no probabilistic rounding
- no adaptive thresholds

Containment is evaluated via direct corner checks:

```
sqrt( (x + dx'_k)^2 + (y + dy'_k)^2 ) <= R
```

B.7 Lattice Membership Verification

Lattice membership is verified exactly.

Axis-aligned lattice:

```
(x - dx) / s ∈ Z
(y - dy) / s ∈ Z
```

Rotated lattice:

```
(u, v) ∈ Z × Z
```

Membership checks are independent of boundary checks.

B.8 Non-Overlap Verification

For rotated-lattice configurations, non-overlap is verified by:

```
min_center_distance >= s
```

This condition is evaluated across all square pairs and recorded explicitly.

B.9 Certification Binding

Each certified configuration includes:

- explicit parameter tuple `(R, s, theta, dx, dy)`
- full square center enumeration
- verified square count `N`
- hash-bound identity over ordered centers

Any alteration to configuration data invalidates the certificate.

B.10 Execution Independence

The execution pipeline does not depend on:

- execution order
- machine timing
- stochastic initialization
- adaptive heuristics

Repeated executions under identical parameters yield identical results.

B.11 Reproduction Requirements

To reproduce any certified result, an independent evaluator requires only:

- the recorded parameters
- the lattice definition
- the boundary verification rule
- the non-overlap condition (when applicable)

No search heuristics or tuning choices are needed.

B.12 Appendix Summary

Appendix B defines the complete experimental configuration under which all FSAE results were generated and verified.

Together with the main document and Appendix A, it provides:

• full transparency
• reproducibility
• verification readiness
• clear execution boundaries

This concludes the formal documentation of the Finite Square Arrangement Exhaustion framework.

OMP