IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 2, July 2026
Impact Factor: 8.2

TextileAI A Deterministic AI Assisted Framework for Weave Design Automation

Yash Nileshbhai Kachhadiya and Mr. Mithilesh Kumar Singh
Student, Department of Computer Science
Professor, Department of Computer Science
Shree Dhanvantary College of Engineering and Technology, Kim Surat, India
yashkachhadiyawork@gmail.com and singhmk2102@gmail.com

Abstract: *Textile weave design converts creative fabric in-tent into loom-executable technical data, including peg plans, draft plans, shaft constraints, yarn parameters, production calculations, and machine-ready instructions. In many small and medium weaving units, this workflow still depends on handwritten design cards, disconnected spreadsheets, and free-lance designer communication. This paper presents TextileAI, a deterministic AI-assisted framework for weave design au-tomation. Artificial intelligence is used for intent interpretation and user interaction, while loom-critical outputs are generated by deterministic textile algorithms for binary matrix gener-ation, shaft-count computation, float validation, production calculation, and export preparation. The prototype is evaluated through source-level inspection, 400 developer-tested design iterations, and a preliminary free-tier field pilot with six textile factories in Surat, India. Observed workflow timings indicate that complete design-card preparation decreased from approx-imately 180 minutes to 10 minutes, fabric calculation from 30 minutes to 4 minutes, and peg-plan/export preparation from 5 minutes to approximately 20 seconds. These findings suggest that bounded AI interaction combined with deterministic man-ufacturing logic can reduce design dependency and improve workflow digitization in intuition-driven textile manufacturing. The evidence remains preliminary and requires controlled user studies, independent expert audit, and machine-specific export validation before industrial generalization.*

Keywords: TextileAI, textile computer-aided design and manufacturing, weave design, shaft-based weaving, determinis-tic generation, AI-assisted design, manufacturing digitization, loom automation

I. INTRODUCTION

Woven fabric design is not only an artistic task; it is a constrained engineering process. A fabric pattern becomes valuable only when it can be represented as a sequence of loom operations [1], [2]. In dobby and computer-aided de-sign and computer-aided manufacturing (CAD/CAM) work-flows, a designer must coordinate warp and weft data, draft plans, lifting or peg plans, shaft capacity, nozzles or feeders, reed and density settings, material parameters, costing, pro-duction rate, and final documentation. In traditional textile clusters such as Surat, India, many of these values are still written manually on design cards and then repeated for loom operators, buyers, sampling records, and internal costing. The same technical design may therefore exist as a sketch, handwritten peg plan, spreadsheet, and verbal instruction, creating a gap between design imagination and production certainty.

The field motivation for this project came from observing how small factories balance cost, speed, and technical dependency. Many firms avoid hiring a full-time designer and instead depend on freelance designers who charge approximately INR 2,000 per design. This reduces fixed salary cost, but it creates a dependency chain: the factory must often book the designer one day in advance, wait for revisions, and sometimes keep a machine idle while a border, peg plan, or draft-related configuration is changed. In such a workflow the main loss is not only design error; it is the loss of production time and design autonomy.

DOI: 10.48175/568

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 2, July 2026
Impact Factor: 8.2

The rise of large language models (LLMs) creates an opportunity to simplify textile design interaction. However, unrestricted AI generation is insufficient for manufacturing. A visually attractive pattern can still be impossible to weave if it exceeds shaft limits, contains excessive floats, lacks interlacement, or cannot be encoded into a controller file. Prior work on human-AI interaction, explainability, and trust in automation emphasizes that intelligent systems should remain understandable, predictable, and controllable in high-consequence workflows [8], [9], [10]. Therefore, a manufacturing-oriented AI textile system must not treat fabric as only an image. It must treat fabric as a physical matrix executed by hardware.

TextileAI was developed around this principle. The sys-tem uses AI for interaction and intent extraction, but it keeps the core textile operations deterministic. The software does not ask the AI model to directly invent binary peg-plan values. Instead, AI output is converted into bounded design parameters, and TypeScript modules compute the real weave matrix, shaft usage, validation warnings, costing values, and export formats. This design follows established human-AI guidance on bounded control and explainability, while remaining tied to the implemented TextileAI prototype [7], [8], [9].

The main research question is how an AI-assisted textile platform can preserve the accessibility of natural-language design while ensuring that every manufacturing-critical out-put is computed by deterministic textile logic.

The contributions of this paper are:

- a hybrid AI-deterministic architecture for textile CAD/CAM design workflows, implemented in Tex-tileAI;
- a matrix-based generation engine for plain, rib, basket, twill, satin, honeycomb, diamond, herringbone, zigzag, crepe, Bedford cord, houndstooth, mock leno, and border-related structures;
- a validation pipeline for shaft count, float length, interlacement integrity, and dobby/jacquard-oriented classification;
- a production-aware calculation layer for EPI, reed space, GSM, yarn consumption, production rate, and cost per meter;
- a digital export layer that supports technical PDF reports and machine-oriented formats including .EP, .WEA, .JC5, .DES, and text peg-plan output;
- a preliminary field-informed evaluation using six free-tier factory pilots and 400 developer-led design tests, with explicit limits on statistical generalization.

The claims in this paper are intentionally bounded. Tex-tileAI is presented as an implemented prototype and field-informed applied-systems contribution, not as a certified industrial controller or a statistically generalized factory study. This boundary is important because manufactur-ing software must be evaluated differently from ordinary web applications: a visually correct interface is insufficient unless the underlying loom instructions remain auditable, reproducible, and controller compatible.

II. PROBLEM DEFINITION

A. Manual Weave Design-Card Workflow

A traditional weave design card contains identity fields, warp and weft material specifications, reed and width cal-culations, peg plan rows, feeder sequences, design notes, and production details. The work is usually performed by a skilled designer or loom technician. The problem is not that the manual method lacks knowledge; the problem is that the knowledge is not stored in a computational form. When the same information is rewritten several times, errors become likely. When a design is modified, earlier paper copies become outdated. When the design must be sent to an electronic loom or CAD/CAM controller workflow, the handwritten lifting plan must be manually converted again.

B. Related Work and Research Gap

TextileAI is positioned at the intersection of textile CAD/CAM, computational weaving, AI-assisted interaction, and manufacturing workflow digitization. Prior computa-tional weaving work has shown that weave structures can be represented as matrix relationships among drawdown, threading, tie-up, and lifting plans [1]. Textile engineering literature also links weave structure to measurable fabric properties, making the matrix representation useful beyond visualization [2]. Industrial tools and interchange formats such as WIF support machine-readable transfer of weav-ing

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 2, July 2026
Impact Factor: 8.2

data [4], while commercial digital weaving tools and machine vendors provide controller-oriented ecosystems for electronic weaving [3], [5], [6].

The gap addressed in this paper is not the invention of weaving mathematics. Rather, the gap is systems integra-tion for local weave-design workflows: AI-assisted input, deterministic weave generation, validation, production cal-culation, design-card documentation, and export preparation are combined in one inspectable pipeline. General AI tools can describe textile concepts, but they do not by themselves provide shaft feasibility, float validation, repeat synchro-nization, or controller-oriented export. Conventional textile CAD tools can be powerful [3]– [6], but they often remain separated from the local design-card and costing practices used by small and medium factories. TextileAI addresses this gap by treating AI as an interface layer and deterministic computation as the manufacturing authority. Table I summa-rizes this positioning against common alternative workflows.

TABLE I: POSITIONING OF TEXTILEAI AGAINST ALTERNATIVE WORKFLOWS

Workflow	Main Limitation	TextileAI Response
Manual designer card	Slow revision and repeated writing	Digital card and instant recalculation
General AI image tool	No loom-feasibility check	Deterministic matrix engine
Conventional CAD	Expert-heavy and often separate from costing	Unified workflow
Current prototype	Needs deeper border validation	Explicit limitation and modular extension

C. Design Requirements

The software was designed around five requirements:

1) Computational validity: generated patterns must be represented as binary matrices and checked before export.
2) Manufacturing awareness: shaft count, float length, repeat size, nozzle usage, and loom capacity must influence the design.
3) Human usability: the user should interact through forms, canvas editing, library browsing, and AI chat rather than only low-level matrix editing.
4) Production linkage: fabric metrics and cost must update from the same design state.
5) Exportability: the final output should be useful for both human operators and electronic machine work-flows.

III. BACKGROUND

A. Binary Interlacement Matrix

Every woven structure can be encoded as a binary inter-lacement matrix P [1], [2]:

$$P \in \{0, 1\}^{R \times C}, \quad (1)$$

where R is the number of picks and C is the number of warp ends in the repeat. In TextileAI, $P(i, j) = 1$ denotes warp over weft at pick i and end j, while $P(i, j) = 0$ denotes weft over warp. This representation is compact, directly visualizable, and suitable for validation.

B. Relationship Between Drawdown and Loom Plan

For shaft weaving, the drawdown can be related to threading, tie-up, and lifting matrices [1]. In Boolean form:

$$P = LUD, \quad (2)$$

where D is the threading matrix, U is the tie-up matrix, and L is the lifting or treadling matrix. Modern electronic loom and CAD/CAM workflows may combine parts of this relationship into a direct lifting plan, but the shaft feasibility constraint remains.

C. Shaft and Machine Feasibility

Shaft-based looms and CAD/CAM workflows control groups of warp ends through shafts [1], [5]. Therefore, warp ends with identical lift sequences can share a shaft, while warp ends with different lift sequences require different shafts. Let Cj be the j th column vector of P. The minimum required shaft count in TextileAI is:

DOI: 10.48175/568

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 2, July 2026
Impact Factor: 8.2

$$S = |\{C1, C2, \ldots, Cn\}|. \quad (3)$$

If S exceeds the target loom capacity, the design is flagged as unsuitable for the specified shaft-based configuration. If the requirement is very high, the design becomes jacquard-oriented.

D. Admissible Design Constraints

For a target shaft-based loom or CAD/CAM workflow with shaft capacity Smax, TextileAI treats a generated matrix as admissible only if it satisfies structural and hardware con-straints derived from matrix-based weaving and machine-readiness requirements [1], [4]. At minimum, the matrix must be binary, shaft-feasible, and structurally interlaced:

$P \in \{0, 1\}R \times C, \; S(P) \leqslant S\text{max}, \; (4)$ and no pick or end may be completely non-interlacing:

The system then applies float constraints using row-wise and column-wise run-length checks. This formulation makes the validation layer inspectable: a rejected design is not simply labeled "bad," but is associated with a specific constraint violation.

IV. SYSTEM ARCHITECTURE

A. Overview

Fig. 1 presents the implemented TextileAI architecture. The system is built as a web application using Next.js, React, TypeScript, Zustand state management, Supabase persistence, and modular textile computation libraries. The AI design assistant uses a Groq-backed chat interface in the current prototype, while deterministic modules perform textile-specific computation.

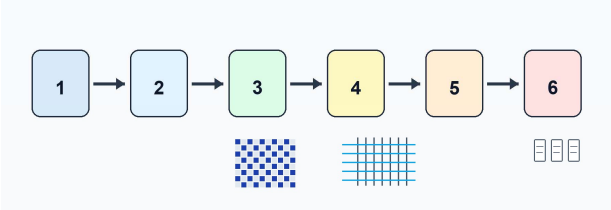

Fig. 1. TextileAI hybrid CAD/CAM architecture. Legend: 1 user input, 2 AI intent bridge, 3 deterministic matrix engine, 4 validation, 5 production calculation, and 6 export package.

B. Module-Level Implementation

The implementation is organized into domain modules rather than one monolithic editor. Table II maps the actual software modules to their research function.

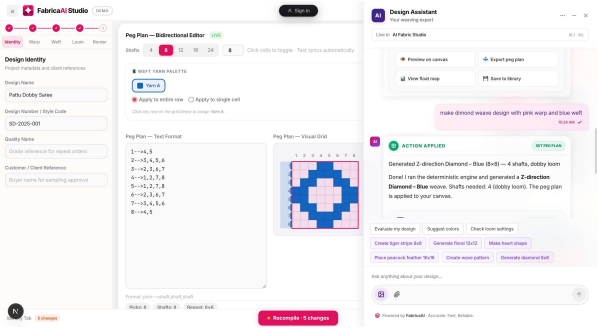

Fig. 2. TextileAI working interface showing the digitized design-card workflow with identity fields, peg-plan text, matrix visualization, and AI assistant output in one workspace.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 2, July 2026
Impact Factor: 8.2

TABLE II: TEXTILEAI SOFTWARE MODULES AND RESEARCH FUNCTIONS

Module	Function in the Framework
weave/ generators.ts	Generates binary matrices for industrial weave families and validates basic matrix properties.
weave/ engine.ts	Builds complete design objects with names, hash IDs, WIF-compatible names, shaft counts, warnings, and metadata.
weave/ massGenerate.ts	Expands the design space through ratios, colors, modifiers, directions, and fabric contexts.
weave/ nlpBridge.ts	Converts user language into deterministic design parameters and peg-plan text.
weave/ draftPlan.ts	Derives draft plans, verifies reconstruction, and prepares WIF-like sections.
border/ engine.ts	Compiles body, side border, and cross-border zones using LCM repeat synchronization and shaft optimization.
calc/ engine.ts	Computes EPI, reed space, GSM, yarn usage, production rate, and cost per meter.
exports/ machineFile.ts	Serializes matrices into .EP, .WEA, .JC5, .DES, and text formats.
store/ designStore.ts	Maintains central design state and coordinates recalculation and Supabase persistence.

C. AI and Deterministic Boundary

The most important architectural decision is the separa-tion between probabilistic reasoning and deterministic man-ufacturing logic, following reliability concerns in human-AI and automation systems [8]– [10]. AI handles user-facing interpretation: for example, recognizing that “make 3/1 twill”, “generate herringbone”, or “create a saree border” refers to a specific textile operation. The deterministic layer then generates the matrix, computes shafts, validates floats, and exports files. This boundary reduces hallucination risk because the model is not trusted to invent loom instructions. It also improves reproducibility: the same parameter set produces the same matrix, validation report, and export output.

TABLE III: AI-DETERMINISTIC BOUNDARY IN TEXTILEAI

Task	AI Layer	Deterministic Layer
Intent extraction	Parses language	Maps to bounded parameters
Weave selection	Suggests family	Calls generator function
Peg-plan values	Does not author final bits	Computes binary matrix
Shaft count	Explains result	Unique-column computation
Float validation	Explains warning	Scans matrix rows/columns
Export	Guides user	Serializes machine file

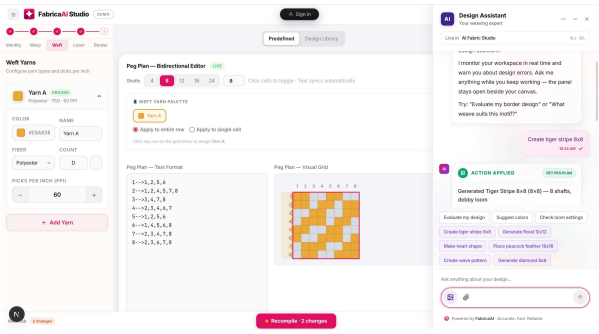

Fig. 3. AI-assisted design generation view. Natural-language input is interpreted by the assistant, while final weave data are applied through deterministic design actions.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 2, July 2026
Impact Factor: 8.2

V. METHODOLOGY

A. Base Matrix Generation

TextileAI generates weave structures through explicit mathematical procedures grounded in matrix-based weave representation [1], [2]. Plain weave is the simplest matrix:

$$P\text{plain} = [1 \;\; 0]\,. \quad (6)$$

For a twill structure with u warp-up and d warp-down positions, the repeat size is $n = u + d$. Direction is controlled by δ, where positive and negative shifts generate opposite twill directions:

Rib and basket weaves are generated through expansion of the plain-weave alternation [1]. Herringbone and zigzag are generated through direction reversal. Satin uses coprime move-number logic:

$$\gcd(n, s) = 1, \;\; s \neq 1, \;\; s \neq n - 1, \;\; (8)$$

where n is the repeat size and s is the satin step. The engine rejects unsupported satin parameter combinations rather than allowing a visually plausible but structurally invalid repeat to pass into export.

This methodology intentionally favors parameterized de-terministic generation over direct pixel-level AI generation.

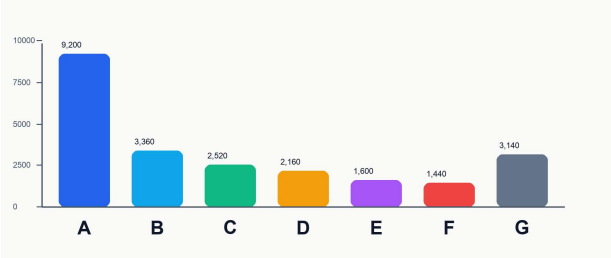

Fig. 4. Estimated parameterized design space represented in TextileAI. Legend: A twill, B crepe, C zigzag, D herringbone, E satin, F broken twill, and G remaining families. Values are code-level candidate counts before filtering.

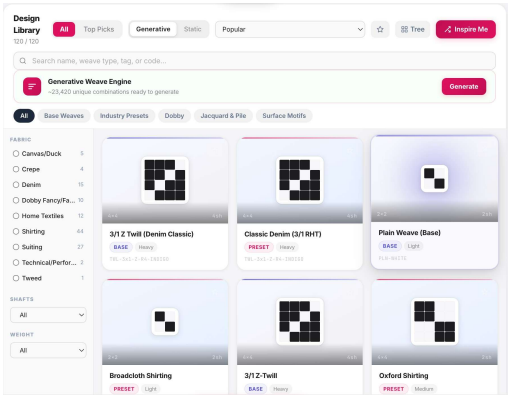

Fig. 5. Design-library interface used to browse and reuse generated weave structures. The interface supports rapid sampling while preserving matrix-level representation for validation and export.

In manufacturing settings, repeatability is a form of engi-neering control: a designer must be able to explain why a design requires a specific shaft count, why a float warning appears, and how the exported lifting plan was derived.

B. Parameterized Design Space

The project does not store thousands of matri-ces manually. It stores generators and parameters. The massGenerate.ts module expands candidate designs through weave ratio, direction, color vector, modifier, and fabric-context combinations. Fig. 4 shows the design-space estimate derived from the implemented generator families. The implemented family-count model gives an estimated 23,420 candidate parameter combinations before UI-level sampling and filtering. This value should be interpreted as a code-level design-space estimate, not as a claim that every combination has been

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 2, July 2026
Impact Factor: 8.2

physically woven or independently approved by textile experts. The largest families are twill, crepe, zigzag, herringbone, satin, and broken twill. This demonstrates that the project scales by combinatorial textile logic rather than by manually drawing each pattern.

C. Shaft Optimization

The draft optimizer assigns identical column sequences to the same shaft, following the shaft-weaving relationship between drawdown and threading [1]. This produces a minimum shaft representation for shaft-based feasibility. Fig. 6 describes the process.

Require: Binary matrix P, maximum shaft count $Smax$
Ensure: Draft array D and shaft count S
1: Initialize empty map M
2: $S \leftarrow 0$
3: for each warp-end column Cj in P do 4: $k \leftarrow$ serialized column pattern of Cj 5: if k is not in M then
6: $M\,[k] \leftarrow S$
7: $S \leftarrow S + 1$
8: end if
9: $D\,[\,j\,] \leftarrow M\,[k]$
10: if $S > Smax$ then
11: return shaft-limit error
12: end if
13: end for
14: return D, S

Fig. 6. Pseudo-code for column-equivalence shaft optimization used for shaft-based feasibility.

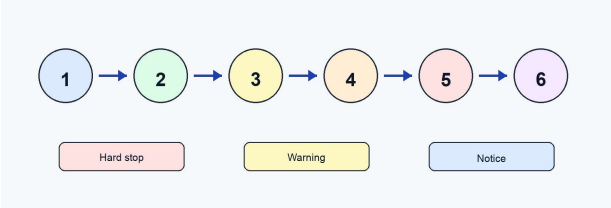

Fig. 7. Validation and manufacturing-readiness pipeline before export. Legend: 1 matrix construction, 2 draft derivation, 3 float scan, 4 integrity check, 5 loom classification, and 6 export packaging.

D. Float and Integrity Validation

Float length is a structural risk because a yarn that passes over too many perpendicular yarns can snag, weaken the surface, or fail during weaving [2]. TextileAI scans vertical runs of 1s for warp floats and horizontal runs of 0s and 1s for weft-side and warp-side float exposure. The maximum run length of value v in sequence A is:

$$F\,(A, v) = \max\{b - a + 1 \mid Ak = v, a \leqslant k \leqslant b\}. \quad (9)$$

$a \leqslant b$

The validation layer also checks all-zero rows, all-one rows, low interlacement ratio, shaft exceedance, and jacquard requirement. Fig. 7 summarizes the pipeline.

E. Border Compilation

Border design introduces an additional challenge because body, side border, and cross-border regions may have dif-ferent repeat lengths. TextileAI synchronizes them through least common multiple (LCM) repeat expansion:

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 2, July 2026
Impact Factor: 8.2

The border compiler expands each zone to the shared repeat, concatenates side zones around the body, derives optimized shafts, and generates a lifting plan. Cross-border logic can switch between body and border cylinders through a pilot sequence. This is important for saree and ethnic fabric workflows where body and border structures cannot be treated as isolated patterns.

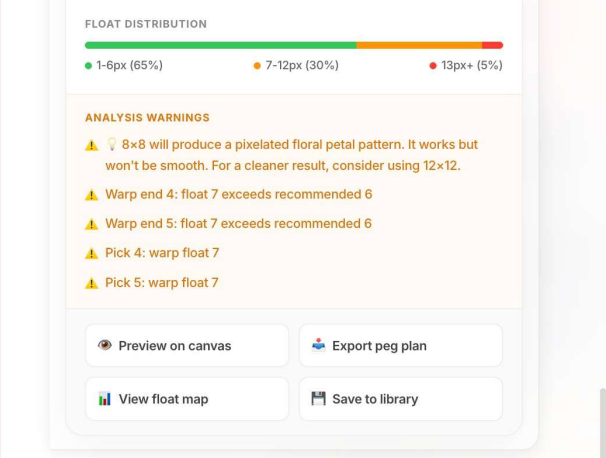

Fig. 8. Example validation-warning interface. The system exposes design risks before export so that a user can adjust parameters before machine preparation.

F. Computational Complexity

Let R be the number of picks, C the number of warp ends, and S the number of unique shaft patterns. Matrix genera-tion for regular weave families is $O(RC)$ because each cell is generated once. Shaft optimization scans each column and compares serialized column patterns, requiring $O(RC)$ time and $O(RS)$ storage for unique column signatures. Float validation scans rows and columns, also requiring $O(RC)$ time. Export serialization is linear in the number of matrix entries or lifted shafts per pick. These bounds are important in practice because they make the system predictable for interactive use: the expensive operations scale with the size of the design matrix rather than with an unbounded search process.

TABLE IV: ALGORITHMIC COST OF CORE TEXTILEAI OPERATIONS

Operation	Time	Main Factor
Matrix generation	$O(RC)$	repeat size
Shaft optimization	$O(RC)$	column signatures
Float validation	$O(RC)$	row/column scans
Export serialization	$O(RC)$	pick encoding
Calculation layer	$O(Y+R)$	yarns and picks

VI. PRODUCTION CALCULATION LAYER

A. Warp and Loom Calculations

The calculation engine links design data to production metrics. Ends per inch (EPI) is computed from Stockport reed count and ends per dent:

$$\text{ReedCount} \times \text{EndsPerDent}$$

$$R\text{global} = \text{lcm}(R\text{left}, R\text{body}, R\text{right}). \quad (10)$$

Reed space is computed from finished cloth width and weft crimp:

DOI: 10.48175/568

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 2, July 2026
Impact Factor: 8.2

ReedSpace = Width × 100 + Crimpweft . (12)
Total warp ends are:
Endstotal = EPI × Width. (13)

B. GSM and Production

TextileAI supports Denier and Ne conversions. The im-plemented Denier-to-Ne conversion is:
Ne = 5315 . (14)
Denier
The GSM estimation is:
GSM = (EPI + PPI) (100 + Crimp) × 0.2327. (15)

C. State and Persistence

The central design state is maintained in a Zustand store in the implemented prototype [7]. Changes to warp, weft, loom, peg plan, and draft sequence trigger recalculation through the calculation engine. Supabase is used for per-sistence so designs can be saved and reloaded. This makes the software more than a single-page calculator; it becomes a versionable design workspace.
Production in meters per hour is:
RPM × 60
Q = PPI × 39.37 × η, (16)
where η is loom efficiency. These calculations are engi-neering estimates whose accuracy depends on yarn count, crimp, loom condition, and local calibration. Their role in the prototype is to keep design, costing, and production planning connected in the same digital design card.

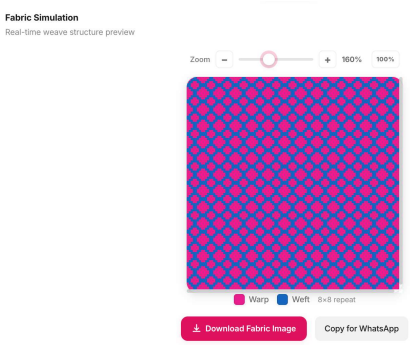

Fig. 9. Fabric simulation view used to inspect the generated structure visually before final reporting or export.

A. Human-Readable Outputs

The system generates technical PDF reports and tradi-tional design-card-style outputs as part of the implemented TextileAI workflow [7]. These outputs organize identity, loom parameters, yarn details, peg plan, simulation preview, production estimates, and costing information in one consis-tent record. This directly addresses repeated manual writing and creates a recoverable design history for later revision.

B. Machine-Oriented Outputs

The machine export module serializes peg-plan matrices into formats such as .EP, .WEA, .JC5, .DES, and plain text. A typical machine-oriented export contains design identity, shaft count or machine classification, pick count, nozzle data, and one row per pick. This provides a bridge from web-based design to controller-oriented file prepara-tion and is aligned with the need for controller-compatible data exchange in modern weaving systems [4]– [6]. The exporter currently represents prototype file preparation; each format should still be validated against the exact controller specification and factory configuration before production deployment.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 2, July 2026
Impact Factor: 8.2

VII. EXPORT AND DIGITAL WORKFLOW

A. Evidence Model

The evaluation follows an applied-systems evidence model rather than a laboratory-only benchmark. Three evidence sources were used. First, the software modules and project documentation were inspected to verify that each claimed function exists in the implemented prototype [7]. Second, 400 designs were generated and iteratively corrected during development to test weave-family cover-age, shaft-count behavior, validation warnings, and export consistency. Third, TextileAI was released in a free-tier pilot to six local textile factories in Surat to observe whether the system reduced waiting time, design dependency, and revision friction in practical workflows.

This evidence model is appropriate for an early CAD/CAM systems paper, but its limits must be stated clearly. The field pilot was not a randomized controlled trial, the 400-design set was developer-led, and timing values are operational observations rather than statistically generalized industry benchmarks. The evaluation therefore supports a claim of promising workflow impact and implemented technical feasibility, not a claim of certified industrial cor-rectness.

TABLE V: DEVELOPER-LED VALIDATION SCOPE

Item	Scope
Design cases	400 iterative design cases
Pattern families	Plain, twill, satin, herringbone, diamond, honeycomb, crepe, basket
Validation checks	Shaft count, float length, impossible rows, matrix consistency
Export checks	Row count, shaft sequence, peg-plan text consistency
Evaluation type	Developer-led prototype validation

B. Implemented Capabilities

Table VI summarizes the current implemented capabili-ties from the project codebase.

C. Workflow Time Comparison

Table VII reports observed approximate workflow times from the manual process and TextileAI-assisted process. The values are based on the project author's field obser-vations and pilot usage. The purpose of this comparison is to estimate order-of-magnitude workflow impact. Percentage reduction is computed as:

$$\text{Reduction} = T\text{manual} - T\text{TextileAI} \times 100. \quad (17)$$

$$T\text{manual}$$

TABLE VI: IMPLEMENTED CAPABILITIES IN TEXTILEAI PROTOTYPE

Capability	Implementation Evidence
Weave generation	Plain, rib, basket, twill, broken twill, herringbone, zigzag, satin, honeycomb, diamond, crepe, Bedford cord, houndstooth, mock leno.
Design-space generation	23,420 candidate parameter combinations estimated from generator families before filtering.
AI design bridge	Natural-language parser maps prompts to deterministic design parameters and peg-plan text.
Shaft/machine feasibility	Unique-column counting and draft derivation classify shaft-based feasibility and jacquard-oriented cases.
Float validation	Matrix scanning detects excessive warp/weft floats and impossible rows.
Production calculations	EPI, reed space, GSM, linear weight, yarn consumption, production, and costing.
Machine export	.EP, .WEA, .JC5, .DES, and text exports.

DOI: 10.48175/568

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 2, July 2026
Impact Factor: 8.2

TABLE VII: PRELIMINARY WORKFLOW TIME COMPARISON

Task	Manual	TextileAI	Reduction
Complete design card with peg plan, draft plan, and connected information	180 min	10 min	94.4%
Fabric/GSM calculation workflow	30 min	4 min	86.7%
Peg-plan/export preparation	5 min	20 sec	93.3%
Design revision or border-related change	designer-dependent	seconds after parameter entry	not separately measured

The complete design-card workflow was reduced from approximately 180 minutes to 10 minutes, an estimated 94.4% reduction. Fabric calculation time reduced from 30 minutes to 4 minutes, an estimated 86.7% reduction. Peg-plan/export preparation reduced from 5 minutes to approx-imately 20 seconds, an estimated 93.3% reduction. These results should be read as preliminary operational evidence. Their importance is practical: when a loom is waiting for a revised design decision, the cost is not only the designer's fee but also idle production time.

D. Cost and Dependency Impact

In the observed workflow, many factories prefer freelance designers because they avoid fixed monthly salary cost. However, the typical freelance cost of approximately INR 2,000 per design and the need to book design work in advance create a dependency bottleneck. TextileAI changes the role of software from a passive documentation tool into an active design-support tool. A factory user can generate, modify, validate, and export a design internally, while still using expert designers for complex or high-value work. This is not a replacement for human expertise; it is a reduction of avoidable waiting time and repeated manual work.

TABLE VIII INTERPRETATION OF EVALUATION EVIDENCE

Evidence	Supported Claim	Remaining Gap
400 developer-led designs	Generator and validation coverage during implementation	Independent audit needed
Six-factory pilot	Practical workflow relevance	Controlled study needed
Timing observations	Order-of-magnitude speed benefit	Statistical confidence needed
Export generation	Prototype file preparation	Controller certification needed

E. Factory Feedback

Six factories used the software in a free-tier pilot. Feedback was qualitatively consistent across three themes. First, users valued speed: design iteration and configuration changes could be attempted without waiting for a freelance designer. Second, users reported improved confidence be-cause validation and calculation were visible in the same workflow. Third, users reported increased autonomy in experimenting with weave structures. This is significant in an industry where much knowledge is intuition-driven and often held by a small number of expert designers or master technicians. Because the feedback was informal, it is used here as field motivation and early usability evidence, not as a statistically validated user study.

F. Benchmark Pattern Coverage

The system was tested against common weave families including plain weave, 2/1 twill, 3/1 twill, 2/2 twill, bas-ket, satin, herringbone, diamond, honeycomb, and crepe. The goal of this benchmark was not to prove physical performance of every generated design, but to verify that the deterministic generator could produce expected matrix structures and that validation logic could compute shaft count and float behavior.

TABLE IX REPRESENTATIVE PATTERN COVERAGE

Pattern Family	Generated	Validation
Plain and basket	Yes	repeat and float check

DOI: 10.48175/568

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 2, July 2026
Impact Factor: 8.2

2/1, 3/1, 2/2 twill	Yes	shaft and direction check
Satin	Yes	coprime step check
Herringbone and diamond	Yes	symmetry and shaft check
Honeycomb/crepe	Yes	float and integrity check

G. Threats to Validity

The evaluation has five main threats to validity. First, the six-factory deployment was a preliminary field pilot rather than a controlled experimental study. Second, workflow times were collected as operational estimates and may vary across designers, loom types, and design complexity. Third, the 400-design validation set was developer-led and has not yet been independently audited by external textile experts. Fourth, machine export formats must be validated against the exact controller and factory configuration before pro-duction use. Fifth, border design is partially implemented, but the effect of border structures on full fabric behavior, yarn usage, and costing requires deeper research. These limitations do not invalidate the system contribution, but they define the boundary of the current evidence.

H. Reproducible Evaluation Plan

To convert the present pilot evidence into stronger conference-level validation, future evaluation should follow a reproducible protocol. Table X summarizes the most important experiments. This plan is included to make the evaluation path explicit and to separate completed evidence from necessary future validation.

TABLE X: RECOMMENDED NEXT-STAGE VALIDATION PROTOCOL

Experiment	Metric	Purpose
Controlled timing study	task time, revision time	Quantify productivity gain
Expert matrix audit	pass/fail, defect class	Verify technical correctness
Controller export test	load success, pick accuracy	Validate machine compatibility
Border fabric study	GSM deviation, balance	Model border effects
User study	usability, confidence	Measure adoption quality

IX. DISCUSSION

The main strength of TextileAI is that it avoids the false simplicity of treating textile design as image generation. Fabric is a physical object produced by machine constraints. TextileAI therefore uses AI as a collaborator at the interface level and deterministic logic at the manufacturing level. This is the difference between a prompt-based drawing tool and an engineering-aware textile CAD/CAM system.

The project is also significant for local textile workflows. Traditional weave design-card preparation contains deep practical knowledge, but that knowledge is not easily search-able, versioned, simulated, or exported. TextileAI preserves the familiar design-card concept while converting it into structured software state. The same data can produce a visual preview, cost estimate, PDF report, and machine file. This matters because many small factories do not lack intelligence or craft skill; they lack accessible digital tools that respect how their work is actually performed.

The field feedback suggests that the value of TextileAI is not only technical automation but also autonomy. When a factory can modify a design internally, it can explore creative alternatives without waiting for a designer's avail-ability. This does not eliminate the role of expert designers; rather, it shifts expert attention toward complex design deci-sions while routine calculation, validation, and reformatting are handled by software.

The current prototype is strongest in shaft-based weave workflows and CAD/CAM-style design-card preparation.

Full jacquard image-to-bitstream generation, additional CAD/CAM controller validation, physical loom testing, and yarn-specific empirical calibration remain future work. Bor-der design is the most important open technical area because border structures affect not only appearance but also fabric balance, yarn usage, and costing. However, the architecture already provides a foundation for these extensions because the internal representation is matrix-based and the export layer is modular.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 2, July 2026
Impact Factor: 8.2

The broader systems lesson is that AI-assisted manu-facturing tools should be designed around accountability. In TextileAI, every AI-assisted action must resolve into a parameter set, every parameter set must produce a matrix, and every matrix must pass explicit validation checks before export. This makes the system inspectable by designers, operators, and future researchers.

X. CONCLUSION

This paper presented TextileAI, a deterministic AI-assisted CAD/CAM framework for weave design automa-tion. The system converts textile design intent into struc-tured parameters, generates binary weave matrices, validates shaft and float feasibility, computes production and costing metrics, and exports human-readable and machine-oriented outputs. A preliminary field-informed evaluation across six factories and 400 iteratively tested designs showed large observed reductions in design-card preparation, fabric cal-culation, and peg-plan/export preparation time. The research contribution is the disciplined boundary between AI in-teraction and deterministic textile computation. By keep-ing loom-critical decisions inside inspectable algorithms, TextileAI provides a practical direction for reliable AI-assisted manufacturing software in an intuition-driven textile industry. Future work will focus on controlled user studies, independent textile expert evaluation, certified machine-controller export testing, and deeper border-design model-ing.

REFERENCES

[1] M. R. Masson and J. Roussel, “A mathematical weaver’s notes and guide to shaft weaving,” University of Arizona Weaving Archive. [Online]. Available: https://www2.cs.arizona.edu/patterns/weaving/
[2] A. H. Nayak, R. Padhye, and L. Wang, “Factors of weave estimation and the effect of weave structure on fabric properties: A review,” Fibers, vol. 10, no. 9, 2022.
[3] Arahne, “Computerized creation of novelty weaves,” technical report. [Online]. Available: https://www.arahne.eu/
[4] WIF Format Organization, “WIF format: Weaving information file specification.” [Online]. Available: https://www.wif-format.org/
[5] Staubli, “Rotary dobby shedding solutions for patterned fabrics,” product documentation. [Online]. Available: https://www.staubli.com/
[6] Picanol, “Weaving machines and machine features,” product docu-mentation. [Online]. Available: https://www.picanol.be/
[7] Y. N. Kachhadiya and M. K. Singh, “TextileAI source code and project documentation,” unpublished academic project, SDCET, Kim, Surat, India, 2026.
[8] S. Amershi et al., “Guidelines for human-AI interaction,” in Proc. CHI Conf. Human Factors in Computing Systems, 2019, pp. 1–13.
[9] A. B. Arrieta et al., “Explainable artificial intelligence (XAI): Con-cepts, taxonomies, opportunities and challenges toward responsible AI,” Information Fusion, vol. 58, pp. 82–115, 2020.
[10] J. D. Lee and K. A. See, “Trust in automation: Designing for appropriate reliance,” Human Factors, vol. 46, no. 1, pp. 50–80, 2004.