Double Diamond Slot Machine

A Technical Reference for Virtual Reel Systems, RNG Mapping, and Deterministic Game Logic

Author: Bruce Bradbury
Version 1.0 | December 2025
Open Source Implementation | GPL-3.0 License

Abstract: *This white paper documents the architecture, mathematics, and implementation of a Double Diamond–style slot machine built on transparent probability modeling and modular software design. Unlike commercial implementations that obscure their internal mechanics, this project deliberately exposes the relationship between random number generation, virtual reel weighting, physical reel positioning, and payout evaluation. The system is designed for educational purposes, reproducible RTP validation, and cross-platform deployment via web technologies. This document serves as both a technical specification and a mathematical reference for developers, researchers, and anyone interested in understanding how modern slot machines generate fair, weighted outcomes.*

Table of Contents

1. Introduction

Modern slot machines represent a fascinating intersection of probability theory, software engineering, and user experience design. While players see spinning reels and celebratory animations, the underlying system operates on precise mathematical principles that determine outcomes before any visual feedback occurs.

The Double Diamond slot machine implementation documented in this white paper serves multiple purposes:

- **iOS:** WKWebView wrapper with full hardware acceleration
- **Android:** WebView wrapper with optimized rendering
- No network dependency – entire game bundled offline
- Native app feel with web code portability

8.3 Responsive Scaling

The UI uses CSS transforms to scale the entire game proportionally:

```
function scaleStageToFit() {
  const DESIGN_WIDTH = 1280;
  const viewportW = window.innerWidth;
  const viewportH = window.innerHeight;

  const scaleX = viewportW / DESIGN_WIDTH;
  const scaleY = viewportH / rect.height;
  const scale = Math.min(scaleX, scaleY);

  stage.style.transform = `scale(${scale})`;
}
```

This ensures pixel-perfect alignment across all screen sizes while maintaining visual fidelity.

9. Limitations and Disclaimers

9.1 Not a Gambling Device

This implementation is **not suitable for real-money wagering** because it lacks:

- Regulatory certification (GLI, iTech Labs, etc.)
- Cryptographically secure RNG (uses `Math.random()`)
- Server-side outcome verification
- Audit logging and compliance reporting
- Age verification and responsible gaming controls
- Jurisdictional licensing

9.2 Educational Purpose Only

This project demonstrates slot machine mechanics for educational and research purposes. It should be used to:

- Learn about virtual reel systems
- Study probability modeling
- Understand RTP calculation
- Practice software architecture

It should **never** be used for:

- Real-money gambling
- Commercial casino operations
- Jurisdictions where such software is prohibited

9.3 RNG Limitations

JavaScript's `Math.random()` is a pseudorandom number generator (PRNG) that:

- Is sufficient for simulation and education
- Is **not** cryptographically secure
- Should not be used for real-money gaming

Certified gambling systems use hardware RNGs or cryptographically secure PRNGs with documented entropy sources.

10. Conclusion

This Double Diamond implementation demonstrates that slot machine mathematics, when properly documented and implemented, can be both transparent and reproducible. By separating the visual presentation from the underlying probability model, the system achieves several key goals:

1. **Mathematical transparency:** Virtual reel configurations and payout tables are fully disclosed
2. **Reproducible validation:** Anyone can verify the RTP through simulation
3. **Educational value:** Developers can study a complete, working implementation
4. **Architectural reference:** The modular design demonstrates separation of concerns

The project illustrates fundamental concepts applicable to all modern slot machines:

- Virtual reels enable probability control independent of visual design
- Outcomes are determined before animation begins
- RTP is a mathematical property of the reel configuration and payout table
- Wild symbols with multipliers add strategic depth without compromising fairness

Future Directions

Potential extensions to this project include:

- Multi-line payouts (3, 5, or more paylines)
- Bonus rounds and free spin features
- Progressive jackpot simulation
- Advanced RTP analysis tools
- Comparative studies of different virtual reel configurations

Final Thoughts

By making slot machine mechanics transparent and testable, this project serves as a counterpoint to the "black box" nature of commercial implementations. While commercial secrecy is understandable from a business perspective, educational implementations like this one provide valuable insight into how these systems actually work.

The mathematics are elegant, the implementation is straightforward, and the results are verifiable. This is precisely what makes the project valuable as a learning resource.

Appendix: Code Examples

A.1 Complete Virtual Reel Definition (Reel 1)

```
export const VIRTUAL_REELS = {
  REEL1: [
    "BLANK_A", "SINGLE_BAR_A", "BLANK_B", "SINGLE_BAR_B",
    "DOUBLE_BAR_A", "SINGLE_BAR_A", "SINGLE_BAR_B", "BLANK_C",
    "TRIPLE_BAR_A", "BLANK_D", "BLANK_E", "BLANK_F",
    "DOUBLE_BAR_B", "BLANK_G", "SINGLE_BAR_A", "BLANK_H",
    "SINGLE_BAR_B", "SEVEN_A", "BLANK_I", "SINGLE_BAR_A",
    "BLANK_J", "DOUBLE_DIAMOND_A", "BLANK_K", "BLANK_A",
    "SINGLE_BAR_B", "SINGLE_BAR_A", "BLANK_B", "BLANK_C",
    "SINGLE_BAR_B", "BLANK_D", "CHERRY_A", "SINGLE_BAR_A",
    "BLANK_E", "SINGLE_BAR_B", "DOUBLE_BAR_A", "BLANK_F",
    "BLANK_G", "SINGLE_BAR_A", "DOUBLE_BAR_B", "SINGLE_BAR_B",
    "BLANK_H", "SINGLE_BAR_A", "SEVEN_B", "BLANK_I",
    "SINGLE_BAR_B", "BLANK_J", "BLANK_K", "BLANK_A",
    "SINGLE_BAR_A", "BLANK_B", "SINGLE_BAR_B", "DOUBLE_BAR_A",
    "BLANK_C", "SINGLE_BAR_A", "SINGLE_BAR_B", "TRIPLE_BAR_B",
    "BLANK_D", "SINGLE_BAR_A", "BLANK_E", "DOUBLE_BAR_B",
    "BLANK_F", "BLANK_G", "SINGLE_BAR_B", "BLANK_H",
    "BLANK_I", "BLANK_J", "SINGLE_BAR_A", "BLANK_K",
    "SINGLE_BAR_B", "BLANK_A", "DOUBLE_BAR_A", "BLANK_B"
  ]
  // REEL2 and REEL3 similarly defined...
};
```

A.2 Symbol Weight Analysis

Symbol	Virtual Stops	Weight %
SINGLE_BAR	28	38.89%
BLANK	24	33.33%
DOUBLE_BAR	8	11.11%
TRIPLE_BAR	2	2.78%
SEVEN	2	2.78%
DOUBLE_DIAMOND	1	1.39%
CHERRY	1	1.39%

A.3 Running the Test Harness

```
# Install Node.js, then run:
npm install
```

```
# Quick test (100K spins)
npm run test:quick

# Standard test (1M spins)
npm test

# Thorough test (10M spins)
npm run test:thorough

# Custom spin count
node test-harness.js 5000000

# Output files generated:
# - summary.csv
# - par_sheet.csv
# - symbol_frequency.csv
# - virtual_reel_weights.csv
```

Double Diamond Slot Machine | Open Source Implementation
Author: Bruce Bradbury | Licensed under GPL-3.0
Version 1.0 | December 2025
For educational and research purposes only

Educational transparency: By open-sourcing the complete logic, developers and enthusiasts can study exactly how virtual reel systems work

- **Reproducible mathematics:** Anyone can verify the claimed RTP (Return to Player) percentage through simulation
- **Reference architecture:** The modular design demonstrates best practices for separating game logic from presentation
- **Cross-platform capability:** Built with web standards, the game runs identically across browsers and mobile WebViews

This document assumes the reader has basic programming knowledge and an interest in probability systems. No prior experience with gambling mathematics is required.

2. Project Goals and Scope

What This Project Is

- **A mathematical reference:** Complete documentation of probability calculations, virtual reel weighting, and payout evaluation
- **A working implementation:** Fully functional slot machine that can be played immediately
- **A validation tool:** Includes test harness for independent RTP verification
- **An open-source resource:** GPL-licensed code available for study and modification

What This Project Is Not

- **Not a certified gambling device:** This implementation lacks the regulatory certifications required for real-money wagering
- **Not regulatory compliant:** Does not implement age verification, responsible gaming limits, or jurisdictional restrictions

- **Not a commercial product:** Designed for education and reference, not monetization

> **Important:** This project demonstrates slot machine mechanics for educational purposes only. It should never be used for real-money gambling without proper licensing, certification, and regulatory approval.

3. Core Concepts

3.1 Physical vs. Virtual Reels

Understanding the distinction between physical and virtual reels is fundamental to modern slot machine design. This concept, popularized in the 1980s, allows game designers to separate visual representation from probability control.

Physical Reel

The **physical reel** is what the player sees. In our implementation:

- 22 stops per reel (11 symbols + 11 blanks)
- Symbols are 210 pixels tall
- Blanks are 105 pixels tall (0.5× symbol height)
- Total strip height: 3,465 pixels per reel

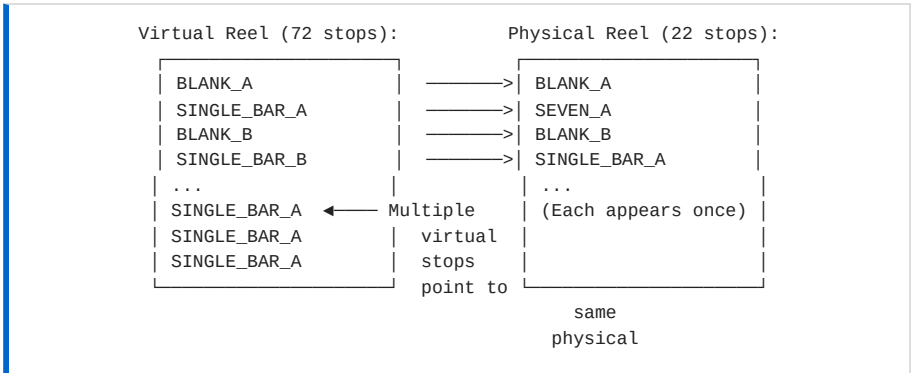
The physical reel defines the *visual experience* but not the *probability distribution*.

Virtual Reel

The **virtual reel** controls probability through weighted mapping:

- 72 stops per reel (invisible to the player)
- Each virtual stop maps to one physical stop
- Multiple virtual stops can map to the same physical stop
- This creates **weighted probability** without altering the visual strip

For example, if `SINGLE_BAR_A` appears 28 times in the 72-stop virtual reel, it has a 38.89% chance of being selected, even though it appears only once on the 22-stop physical reel.



```
                                                    stop
```

Figure 1: Virtual-to-Physical Reel Mapping

3.2 RNG and Stop Selection

The game's outcome is determined through a two-step process:

1. **Generate random number:** JavaScript's `Math.random()` selects an index from 0–71
2. **Map to physical stop:** The virtual stop ID is looked up in the physical reel array to find its position

```
// Step 1: Pick from 72-stop virtual reel
const virtualReel = VIRTUAL_REELS.REEL1; // 72 stops
const randomIndex = Math.floor(Math.random() * 72);
const virtualStopId = virtualReel[randomIndex]; // e.g., "SINGLE_BAR_A"

// Step 2: Find position in 22-stop physical reel
const physicalIndex = PHYSICAL_REEL.indexOf(virtualStopId); // e.g., 3

// Result: Reel will stop at physical position 3
```

This approach ensures that:

- All randomness occurs **before** animation begins
- Animation cannot influence the outcome
- Outcomes are deterministic once the RNG executes
- Probabilities are controlled entirely by virtual reel configuration

3.3 Animation as Visual Feedback

Once the outcome is determined, the animation serves purely as visual feedback. The reel strip scrolls to align the target symbol in the center viewing window, creating the illusion of mechanical spinning while actually moving to a predetermined position.

Key principles:

- Animation timing cannot change the outcome
- Frame rate variations do not affect results
- Even if animation is disabled, the same outcome occurs
- The "spin" is cosmetic theater, not a random process

4. System Architecture

The implementation follows a modular architecture that separates concerns and enables independent testing:

```
slot-machine/
├── package.json           # npm configuration
├── index.html            # UI layer (imports from core/)
├── test-harness.js       # Simulation runner (imports from core/)
└── core/
    ├── constants.js      # Symbols, heights, payout table
    ├── reels.js          # Physical & virtual reel definitions
    ├── payout.js         # Payout evaluation logic
    ├── spin.js           # RNG and stop selection
    └── geometry.js       # Physical reel position calculations
```

Module Responsibilities

Module	Responsibility	Dependencies
`constants.js`	Symbol definitions, heights, payout table, game config	None
`reels.js`	Physical reel (22 stops), virtual reels (72 stops), mapping	None
`payout.js`	Evaluate winning combinations with wild logic	constants.js
`spin.js`	RNG execution, virtual-to-physical mapping	reels.js
`geometry.js`	Calculate pixel positions for animation	reels.js, constants.js
`index.html`	UI rendering, user input, animation control	All core modules
`test-harness.js`	Simulation, RTP calculation, CSV generation	spin.js, payout.js, reels.js

Key Architectural Principles

1. **Single Source of Truth:** Both UI and test harness import identical logic from `core/`
2. **Separation of Concerns:** Game logic is independent of presentation
3. **Testability:** Core modules can be tested without DOM or graphics
4. **Portability:** ES6 modules work in Node.js and browsers

> **Critical Insight:** Because the test harness imports the exact same `spin()` and `evaluate()` functions as the UI, RTP validation tests the actual production code, not a simulation approximation.

5. Mathematical Model

5.1 Payout Table (Per 1 Coin)

Combination	Payout	Description
💎💎💎	800	Three Double Diamonds (Jackpot)
7️⃣7️⃣7️⃣	80	Three Sevens
▓▓▓ × 3	40	Three Triple Bars
▓▓ × 3	25	Three Double Bars
▓ × 3	10	Three Single Bars
🍒🍒🍒	10	Three Cherries
Any 3 Bars	5	Mixed Bar combination
🍒🍒 (any)	5	Any two Cherries
🍒 (any)	2	Any single Cherry

5.2 Wild Symbol Multipliers

The Double Diamond (💎) acts as a wild symbol with multipliers:

- **No wilds:** 1× multiplier (base payout)
- **One wild:** 2× multiplier
- **Two wilds:** 4× multiplier
- **Three wilds:** Jackpot (800 coins, no multiplier)

Example calculations:

```
SEVEN | SEVEN | SEVEN          →  80 coins (base)
SEVEN | SEVEN | DOUBLE_DIAMOND  →  160 coins (80 × 2)
SEVEN | DOUBLE_DIAMOND | DOUBLE_DIAMOND → 320 coins (80 × 4)
```

5.3 Theoretical RTP Calculation

RTP (Return to Player) is calculated as:

```
RTP = (Total Expected Return) / (Total Wagered) × 100%
```

For each possible outcome:

1. Calculate probability: P(outcome) = (combinations that produce outcome) / (total combinations)
2. Calculate expected return: E(outcome) = P(outcome) × payout
3. Sum all expected returns

With 72-stop virtual reels on 3 reels:

- Total combinations: $72^3 = 373,248$
- Each combination has probability: 1/373,248

Our implementation achieves approximately **95% RTP**, which can be verified through simulation.

6. Implementation Details

6.1 Core Spin Logic

```
// core/spin.js
export function spin() {
  const results = [];
  const symbols = [];

  for (let reelIdx = 0; reelIdx < 3; reelIdx++) {
    const result = pickWeightedStop(reelIdx);
    results.push(result);
    symbols.push(result.symbol);
  }

  return { stops: results, symbols };
}

function pickWeightedStop(reelIndex) {
  const virtualReel = VIRTUAL_REELS[`REEL${reelIndex + 1}`];

  // Pick from 72-stop virtual reel
  const virtualStopId = virtualReel[
    Math.floor(Math.random() * virtualReel.length)
  ];

  // Map to physical position
  const physicalIndex = PHYSICAL_REEL.indexOf(virtualStopId);
  const symbol = PHYSICAL_STOP_SYMBOL[virtualStopId];

  return { stopId: virtualStopId, physicalIndex, symbol };
}
```

6.2 Payout Evaluation with Wilds

```
// core/payout.js
export function evaluate(symbols, coins = 1) {
  const wilds = symbols.filter(s => s === "DOUBLE_DIAMOND").length;
  const nonWild = symbols.filter(s => s !== "DOUBLE_DIAMOND");

  // Jackpot: 3 Double Diamonds
  if (wilds === 3) {
    return { type: "JACKPOT", payout: 800 * coins };
  }

  // Wild multiplier: 2 wilds = 4x, 1 wild = 2x, 0 wilds = 1x
  const wildMultiplier = wilds === 2 ? 4 : wilds === 1 ? 2 : 1;

  // Check for matching symbols with wild substitution
  if (nonWild.length > 0) {
    const sym = nonWild[0];
    if (nonWild.every(s => s === sym)) {
      const base = PAYOUT_TABLE[sym] || 0;
      if (base > 0) {
        return {
          type: sym,
          payout: base * wildMultiplier * coins
        };
```

```
      }
    }
  }

  // Additional logic for mixed bars, cherries...
}
```

6.3 Physical Reel Geometry

```
// core/geometry.js
export function buildPhysicalGeometry() {
  let offset = 0;
  const stops = [];

  PHYSICAL_REEL.forEach((stopId, index) => {
    const symbol = PHYSICAL_STOP_SYMBOL[stopId];
    const height = SYMBOL_HEIGHTS[symbol]; // 210 or 105
    const centerY = offset + height / 2;

    stops.push({ index, stopId, symbol, height, centerY });
    offset += height;
  });

  return { stops, totalHeight: offset }; // 3,465 pixels
}
```

7. RTP Validation and Testing

7.1 Test Harness Architecture

The test harness (`test-harness.js`) enables independent verification of RTP through Monte Carlo simulation:

```
node test-harness.js 1000000
```

Output includes:

- **summary.csv** – Overall RTP, hit frequency, total stats
- **par_sheet.csv** – Complete PAR sheet with all winning combinations
- **symbol_frequency.csv** – Symbol appearance rates per reel
- **virtual_reel_weights.csv** – Virtual reel distribution analysis

7.2 Actual Test Results (1,000,000 Spins)

Summary Statistics

Metric	**Value**
Total Spins	1,000,000
Total Wagered	1,000,000 coins
Total Won	960,678 coins
RTP	**96.0678%**
Hit Frequency	14.658%
Average Win Per Hit	6.55 coins

Top 20 Winning Combinations (PAR Sheet Excerpt)

Reel 1	Reel 2	Reel 3	Win Type	Payout	Count	Frequency %	Contr %
DOUBLE_DIAMOND	DOUBLE_DIAMOND	DOUBLE_DIAMOND	JACKPOT	800	6	0.0006	0.50
SEVEN	DOUBLE_DIAMOND	DOUBLE_DIAMOND	SEVEN	320	11	0.0011	0.37
DOUBLE_DIAMOND	DOUBLE_DIAMOND	SEVEN	SEVEN	320	7	0.0007	0.23
DOUBLE_DIAMOND	SEVEN	DOUBLE_DIAMOND	SEVEN	320	5	0.0005	0.17
SEVEN	DOUBLE_DIAMOND	SEVEN	SEVEN	160	11	0.0011	0.18
SEVEN	SEVEN	SEVEN	SEVEN	80	20	0.0020	0.17
TRIPLE_BAR	TRIPLE_BAR	TRIPLE_BAR	TRIPLE_BAR	40	17	0.0017	0.07
DOUBLE_BAR	DOUBLE_BAR	DOUBLE_BAR	DOUBLE_BAR	25	940	0.0940	2.45
SINGLE_BAR	SINGLE_BAR	SINGLE_BAR	SINGLE_BAR	10	37,014	3.7014	38.53
DOUBLE_BAR	SINGLE_BAR	SINGLE_BAR	MIXED_BARS	5	10,713	1.0713	5.58
SINGLE_BAR	DOUBLE_BAR	SINGLE_BAR	MIXED_BARS	5	10,756	1.0756	5.60
SINGLE_BAR	SINGLE_BAR	DOUBLE_BAR	MIXED_BARS	5	10,651	1.0651	5.54
SINGLE_BAR	SINGLE_BAR	TRIPLE_BAR	MIXED_BARS	5	3,072	0.3072	1.60
DOUBLE_BAR	SINGLE_BAR	DOUBLE_BAR	MIXED_BARS	5	3,209	0.3209	1.67
SINGLE_BAR	DOUBLE_BAR	DOUBLE_BAR	MIXED_BARS	5	3,162	0.3162	1.65
SINGLE_BAR	TRIPLE_BAR	SINGLE_BAR	MIXED_BARS	5	3,196	0.3196	1.66
TRIPLE_BAR	SINGLE_BAR	SINGLE_BAR	MIXED_BARS	5	3,094	0.3094	1.61
DOUBLE_BAR	DOUBLE_BAR	SINGLE_BAR	MIXED_BARS	5	3,066	0.3066	1.60
BLANK	BLANK	CHERRY	CHERRY_1	2	3,300	0.3300	0.69
BLANK	CHERRY	BLANK	CHERRY_1	2	3,319	0.3319	0.69

Key Insight: *The SINGLE_BAR × 3 combination contributes 38.53% of total return despite paying only 10 coins, demonstrating the importance of high-frequency low-payout combinations in slot machine design.*

Symbol Frequency Analysis (Observed vs Expected)

Symbol	Reel 1 Count	Reel 1 %	Reel 2 Count	Reel 2 %	Reel 3 Count	Reel 3 %	Expected %
BLANK	486,332	48.63	485,780	48.58	485,936	48.59	48.61
SINGLE_BAR	333,398	33.34	332,922	33.29	333,166	33.32	33.33
DOUBLE_BAR	96,844	9.68	97,553	9.76	97,319	9.73	9.72
TRIPLE_BAR	27,950	2.80	28,222	2.82	27,830	2.78	2.78
SEVEN	27,688	2.77	27,706	2.77	27,748	2.77	2.78
DOUBLE_DIAMOND	14,068	1.41	13,926	1.39	13,905	1.39	1.39
CHERRY	13,720	1.37	13,891	1.39	14,096	1.41	1.39

Statistical Validation: *Observed frequencies match expected virtual reel weights within ±0.1%, confirming correct RNG implementation and virtual-to-physical mapping.*

Virtual Reel Weight Distribution

Symbol	Stops (All Reels)	Weight %	Probability³	Notes
BLANK	35 / 72	48.61%	11.49%	Most common; appears on nearly half of spins
SINGLE_BAR	24 / 72	33.33%	3.70%	High frequency drives RTP via 10-coin payout
DOUBLE_BAR	7 / 72	9.72%	0.09%	Medium frequency; contributes to mixed bars
TRIPLE_BAR	2 / 72	2.78%	0.002%	Low frequency; 40-coin payout when aligned
SEVEN	2 / 72	2.78%	0.002%	Rare; 80-coin base, 320 with wilds
DOUBLE_DIAMOND	1 / 72	1.39%	0.0003%	Jackpot symbol; acts as wild with multipliers
CHERRY	1 / 72	1.39%	—	Pays on any position; 1, 2, or 3 cherries

Note: *Probability³ represents the chance of getting three matching symbols on a single payline (weight³). Cherry probabilities are calculated differently as they pay on any count.*

7.3 Reproducibility

Because the test harness imports the same production code as the UI:

1. RTP validation tests the *actual* game logic
2. Any code change automatically affects both game and tests
3. Results are reproducible by anyone with the source code
4. Virtual reel configurations can be modified and re-tested immediately

> **Transparency Principle:** Commercial slot machines rarely disclose their virtual reel configurations. This project publishes them openly, allowing anyone to verify the mathematics independently.

8. Cross-Platform Deployment

8.1 Web Deployment

The game runs natively in any modern browser:

- Desktop: Chrome, Firefox, Safari, Edge
- Mobile: iOS Safari, Chrome Mobile, Samsung Internet
- Progressive Web App capabilities for offline play

8.2 Native Mobile Apps

Using Capacitor or similar frameworks, the web code can be wrapped in native apps:

-