CuttingEdge AI

Reinforcement Learning for Unique 2-Dimensional Cutting Stock Problems

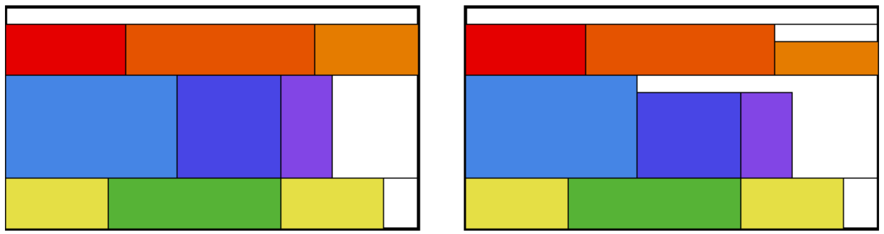
Greg Bateham, Sunim Acharya, Satyam Thakur, Joyce Champie,
Anastasiya Dmytryk

The Problem: Fabric Waste in Fashion

- Over 10 to 25% of fabric is thrown away in large fabric manufacturing industries every day, leading to significant material waste and higher spending costs for industries.
- The 2-Dimensional Cutting Stock Problem has been introduced in multiple ways to attempt to utilize the maximum amount of fabric from each piece of cloth to combat these numbers.
- Despite this, there still lacks an optimal solution for pieces of cloth that do not meet the criteria due to irregular shapes, defects, and directional constraints.
- Our goal is to allow for the possibility of any shape of fabric to not only be used but optimized for the maximum amount of waste reduction.

Why 2DCSP is Difficult

- Irregular pattern pieces and fabric boundaries make it difficult to predict dimensions.
- Directional constraints (nap, print alignment)
- Fabric defects that invalidate placement regions
- NP-Hard combinatorics due to potential rotation, defects, and overlap
- Traditional heuristics often converge on sub-optimal solutions

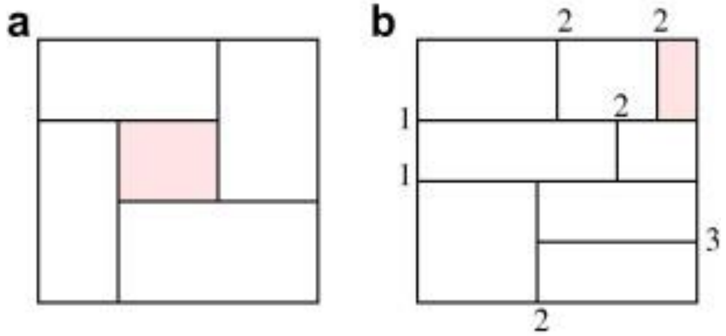


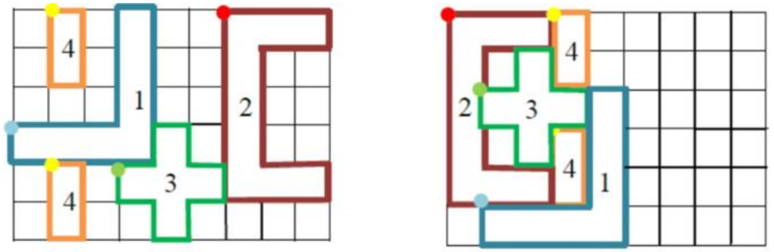
24×10

(a)

24×10

(b)





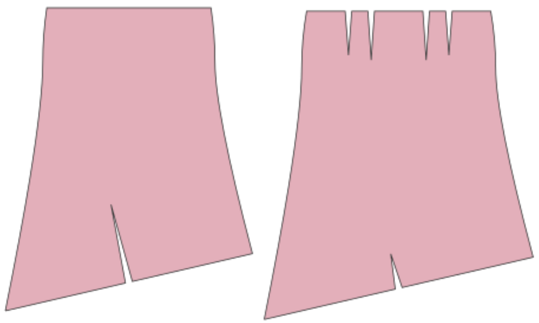
(a) A feasible solution

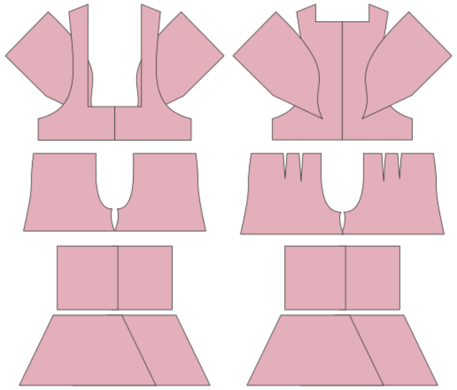
(b) An optimal solution

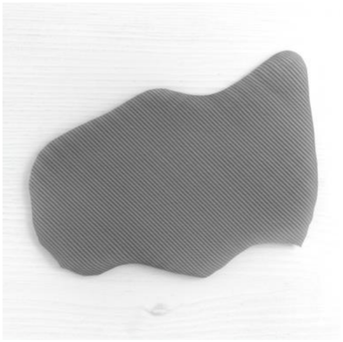
Related Works

- •Turan ’24 – LSTM improves corner-point detection for reliable pattern shape recognition.
    - Provides reliable contour extraction for our PatternRecognitionModule
- •Huang et al. ’23 – Hierarchical RL tackles irregular object packing with multiview heightmaps.
    - Helped our curriculum strategy & multi-stage reward shaping
- •Korosteleva et al. ’22-24 – CNN/GAN pipelines (GarmentCode, NeuralTailor) advance garment feature extraction.
    - Provided domain-specific datasets for our learning pattern
- •Shi et al. ’24 – Tree-Search RL handles complex industrial constraints in 2D cutting stock optimization.
    - Validated our choice to use RL + search hybrid for complex cutting tasks
- •Retondaro & Esperança ’22 – 2D Ball Trees speed curved-shape queries, aiding cloth boundary detection.
    - Provided a potential lightweight alternative for enhancing cloth boundary detection in our segmentation module

Our Approach







- Computer Vision (CV) extracts precise cloth and pattern parameters
- Reinforcement Learning (PPO) learns optimal placement strategies
- Fallback heuristics ensure robustness when RL cannot place any remaining pieces
- Modular pipeline enables incremental improvements and easy debugging

Dataset:

- Over 23500 different garment models; over 400 GB worth of garment mesh segmentation data, images, and other parameter-related files (.json, .yaml, etc)
- Each garment represents a unique design
- 7-8 different randomly-generated cloth patterns (Google Gemini)

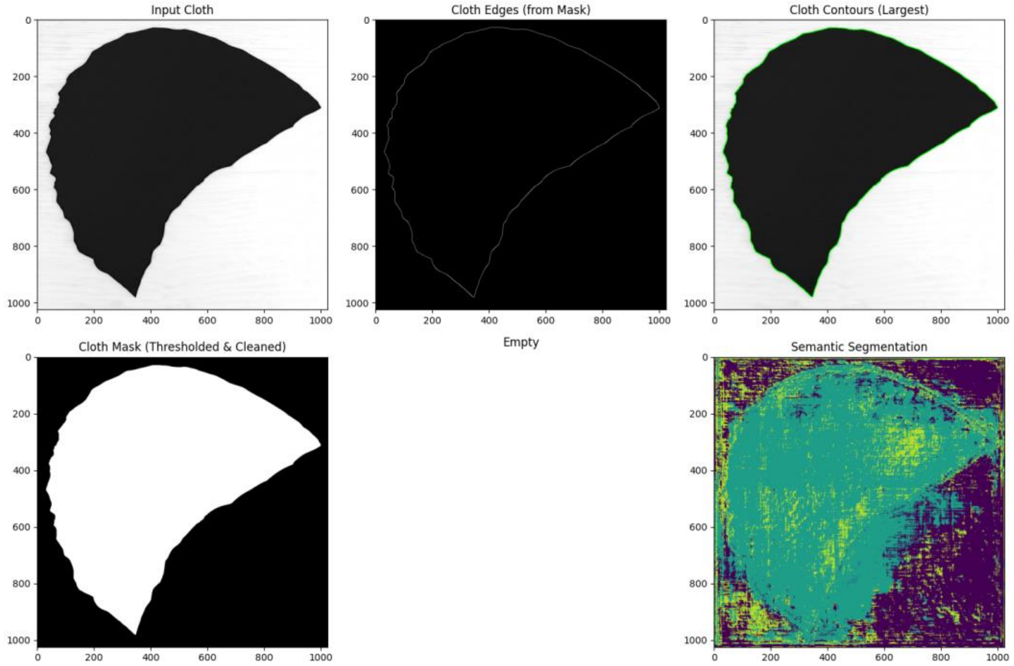
End-to-End Workflow Pipeline

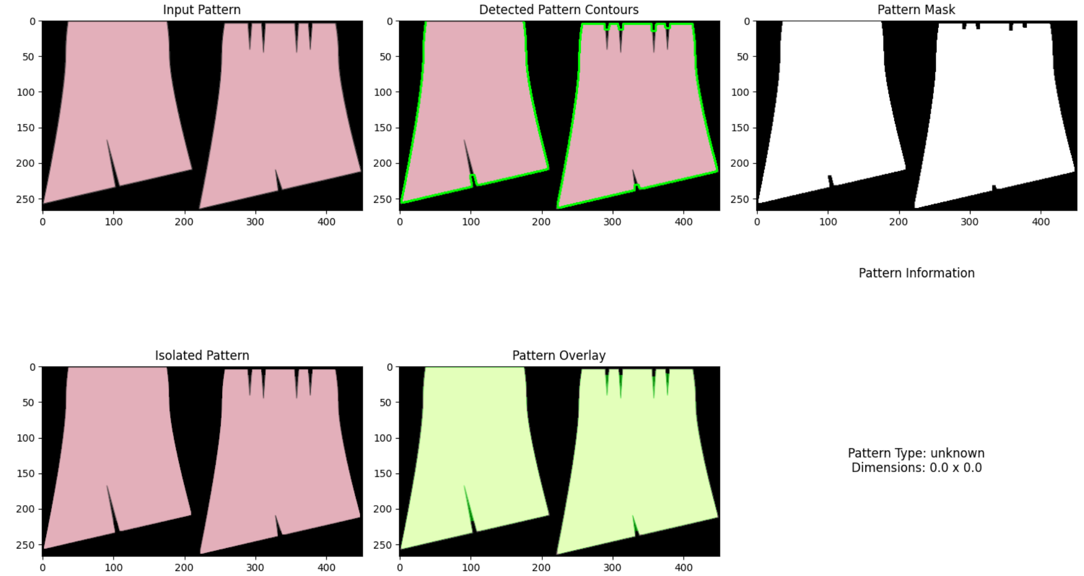
Stage	Technique/Library	Output
ClothRecognitionModule	- OpenCV: Otsu threshold, morphology - Shapely: polygon ops	Cloth boundary polygon, width/height
PatternRecognitionModule	- PyTorch ResNet50 feature extractor - segmentation_models_pytorch U-Net	Contours, dimensions, feature vectors for each pattern piece
PatternFittingModule	- PPO agent (stable_baselines3) - Heuristic fallback + Shapely overlap checks	Placement list (x,y,θ), layout grid, utilization %
Metrics & Visualization	NumPy, Matplotlib / seaborn	Summary CSV, utilization chart, diagnostic plots

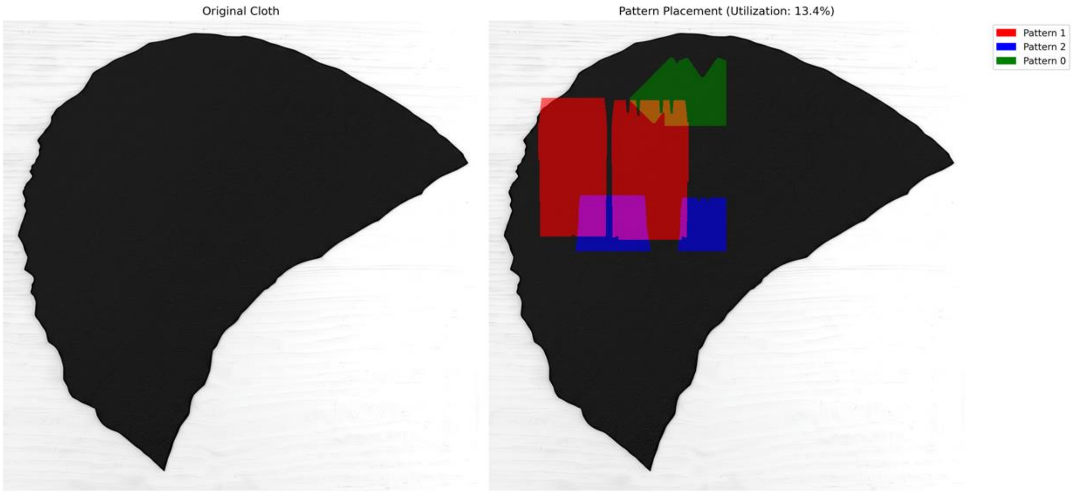
Reinforcement Learning Agent (PPO)

- Environment: Discrete grid representation of cloth; observation = current occupancy grid + next pattern dimensions.
- Objective: Maximize incremental fabric utilization clip range $\varepsilon = 0.2$ guards against large policy updates.
- Action: (x, y, rotation) placement proposal
- Reward: +Utilization gain, -Overlap penalty
- Training: 2048 steps/rollout, $\gamma = 0.99$, GAE $\lambda = 0.95$
- 4 epochs per update, minibatch size of 64
- Early phase curriculum: start with single-pattern episodes -> gradually increase pattern count.
- Fallback: If PPO fails to place a piece after N attempts, PatternFittingModule triggers heuristic_spiral_search().

Key Results


Input Cloth
Cloth Edges (from Mask)
Cloth Contours (Largest)
Cloth Mask (Thresholded & Cleaned)
Empty
Semantic Segmentation


Input Pattern
Detected Pattern Contours
Pattern Mask
Pattern Information
Isolated Pattern
Pattern Overlay
Pattern Type: unknown
Dimensions: 0.0 x 0.0


Original Cloth
Pattern Placement (Utilization: 13.4%)
Pattern 1
Pattern 2
Pattern 0

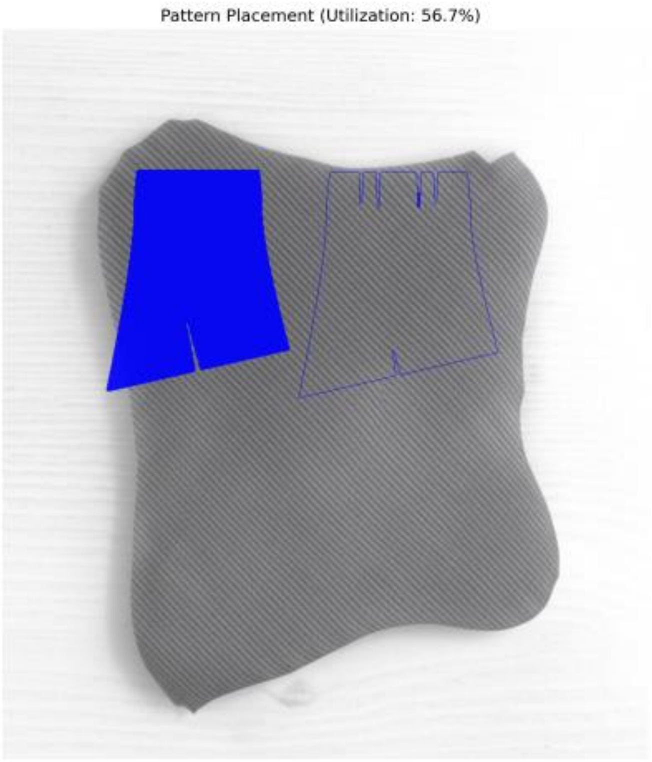
Overall Results

- Hybrid CV+RL pipeline surpasses rule-based heuristics for irregular patterns.
- Data augmentation & domain randomization are vital for real-world generalization.
- RL training benefits from curriculum learning: start with single patterns, scale up.
- Heuristic fallback dramatically reduces invalid placements in early RL episodes.

Metric Category	Parameter	macOS (CPU)	Linux (CUDA)	Notes
System	Processing Device	CPU	CUDA	Platform-specific processing
	Framework Version	Python 3.13	Python 3.12	Version difference noted
Input	Patterns Processed	3	1	Variation in batch size
	Recognition Model	ResNet50	ResNet50	Consistent model architecture
	Pattern Type	Unknown	Unknown	Similar recognition results
Cloth	Dimensions	977 × 962	977 × 962	Identical input size
	Total Area (pixels)	506,396	506,396	Consistent area measurement
	Image Format	JPEG	JPEG	Standard format used
Output	Patterns Placed	3	1	Different placement count
	Utilization Rate	1.01%	10.38%	Higher efficiency on CUDA
	Placement Method	Advanced Manual	Advanced Manual	Consistent methodology
	Model Status	Manual Fallback	Manual Fallback	Similar fallback behavior

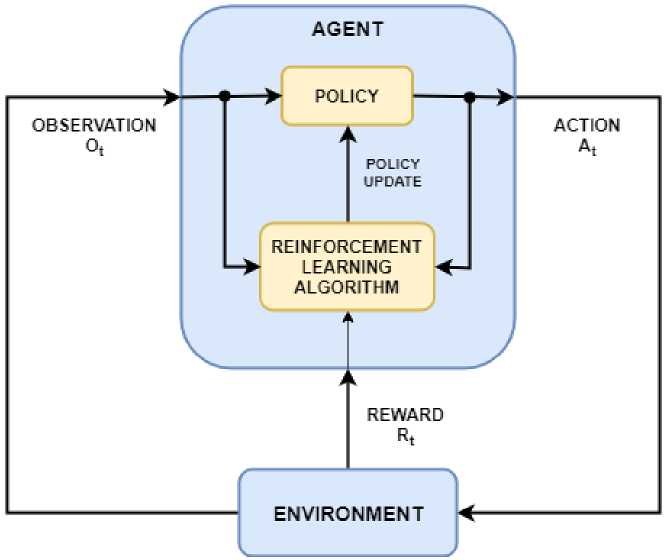
Challenges Encountered

- Image Quality Variations (lighting, shadows) affected contour detection
- Balancing exploration vs. Exploitation in PPO required hyperparameter tuning
- High compute demand for RL
- Handling fabric defects and directionality in real samples remains non-trivial.



Future Directions

- Integrate defect & directionality detection into preprocessing module
- Experiment with Vision Transformers for richer feature extraction
- Evaluate multi-agent RL or SAC for faster convergence.
- Deploy a user-friendly GUI for manual override and real-time visualization.
- Benchmark on industry-standard datasets to validate generalization.



Impact and Conclusion

- Having an automated AI pipeline reduces fabric waste, lowering costs and boosting sustainability.
- Framework is modular – adaptable to other irregular packing tasks (e.g., sheet metal, wood, glass)
- Sets the groundwork for future research in AI-driven textile manufacturing.

Questions?