Controllable Mathematical Reasoning via Self-Optimizing Thought Vectors

Xuying Li
xuyingli.kepler@gmail.com

Abstract

We present a novel approach for controllable mathematical reasoning that leverages self-optimizing thought vectors with entropy minimization. Our method introduces learnable thought vectors that dynamically modulate the internal reasoning process of large language models. Using Gemma-2-9B on GSM8K, we achieve 90.1% accuracy with a controllability score of 0.42, demonstrating that entropy-based rewards effectively guide focused reasoning patterns without requiring external reward annotations. Our analysis reveals distinct thought vector clusters and consistent low-entropy distributions across control conditions, validating our framework for controllable AI reasoning.

1 Introduction

Current large language models excel at mathematical reasoning but lack fine-grained controllability over their internal reasoning processes. While existing approaches like CTRL [1] use control codes prepended to inputs and PPLM [2] manipulates hidden states during generation, they cannot directly influence how models think internally. We propose a fundamentally different approach: controlling internal thought processes through learnable thought vectors.

Our key insight is that mathematical reasoning can be viewed as a selection process among different computational pathways. Consider solving "If John has 23 apples and gives away 8, how many remain?" A model might activate a "direct arithmetic" thought vector for simple subtraction, while for "John has 23 apples, gives 8 to Mary and 5 to Tom, how many remain?" it might blend "multi-step tracking" and "sequential subtraction" vectors. By introducing thought vectors that represent these patterns and using entropy as a measure of focused thinking, we can guide models toward more controlled reasoning. Unlike chain-of-thought prompting [3] which only affects output format, our method modulates internal representations.

We make four primary contributions: (1) A novel architecture featuring self-optimizing thought vectors with entropy-based optimization, eliminating external reward requirements; (2) A three-dimensional control framework for reasoning depth, length, and path; (3) Empirical validation achieving 90.1% accuracy on GSM8K [4]; (4) Comprehensive analysis revealing how thought vectors organize into meaningful patterns.

2 Related Work

Controllable Generation. CTRL [1] introduced control codes for conditional text generation with a 1.63B parameter transformer. PPLM [2] uses gradient-based steering without model retraining. Recent work includes GeDi [5] and FUDGE [6]. However, these methods focus on output-level control rather than internal reasoning processes.

Mathematical Reasoning. Chain-of-thought prompting [3] dramatically improves mathematical reasoning by encouraging step-by-step solutions. GSM8K [4] provides a benchmark for grade-school math problems. Recent approaches include self-consistency [7] and tree-of-thoughts [8]. Our work builds on these while adding explicit control dimensions.

Reward-based Training. RLHF [9] and constitutional AI [10] use human feedback for alignment. Our entropy-based approach is self-supervised, deriving rewards from internal reasoning structure without annotations.

3 Methodology

3.1 Thought Vector Architecture

Our system maintains eight learnable thought vectors, each representing distinct reasoning strategies:

- t_1-t_2**: Direct Computation** - Simple arithmetic operations, fact retrieval
- t_3-t_4**: Sequential Tracking** - Multi-step calculations, running totals
- t_5-t_6**: Algebraic Reasoning** - Variable manipulation, equation solving
- t_7-t_8**: Verification/Checking** - Answer validation, unit consistency

These vectors are initialized orthogonally to ensure diversity. When processing a problem like "A store has 45 items, sells 60% on Monday and half the remainder on Tuesday," the model might strongly activate t_3 (sequential tracking) and t_2 (arithmetic), while keeping others dormant. The selection mechanism uses the current hidden state to compute attention weights over thought vectors, creating a weighted combination that represents the active reasoning approach.

The integration with pre-trained models uses a gating mechanism that learns how much influence thought vectors should have at each position. When confident, the gate strongly activates thought-enhanced representations; otherwise, it preserves original hidden states. This selective activation is learned entirely from data without explicit supervision.

3.2 Control Framework

We operate in three carefully chosen dimensions: **Depth** (1-5) controls reasoning complexity from direct calculations to multi-step derivations; **Length** (2-6) determines solution verbosity; **Path** (binary) selects between direct computation and step-by-step reasoning. These signals are encoded through a deep network with residual connections, progressively transforming the 3D control vector into a 4096D representation that modulates thought vector selection.

3.3 Entropy-Based Self-Optimization

Our most innovative contribution is using entropy minimization as a self-supervised training signal. The entropy of thought vector selection distribution measures reasoning focus: low entropy indicates confident strategy selection, while high entropy suggests uncertainty. By rewarding low entropy during training ($\mathcal{R} = -H(p)$), we encourage decisive reasoning patterns without external supervision.

We implement this through reward-augmented supervised learning, simultaneously optimizing standard cross-entropy loss and entropy-based rewards with weight $\lambda = 0.1$. This creates a virtuous cycle where clearer thinking leads to better results, reinforcing successful thought vector combinations.

4 Experiments

4.1 Setup

We evaluate on GSM8K [4] using Gemma-2-9B [11] as our base model. Through LoRA adaptation [12] and thought vector components, we add 1.06B trainable parameters. Training uses batch size 1 with gradient accumulation over 8 steps, learning rate 2e-5, for 1000 steps. Thought vectors are injected at layer 21, chosen through preliminary experiments.

Table 1: Performance comparison on GSM8K test set

Model	Accuracy	Ctrl	Entropy	Avg Length
Gemma-2-9B	0.211	-	-	1365
+ Chain-of-Thought [3]	0.7725	-	-	1039
+ Supervised Fine-Tuning	0.845	-	-	981
+ Our Method	**0.901**	**0.417**	**4.228**	**976**

4.2 Main Results

Our method achieves the highest accuracy while introducing controllability. The base model performs poorly (21.1%), while chain-of-thought prompting reaches 89.7%. Our approach surpasses both at 90.1% while adding controllable reasoning capabilities.

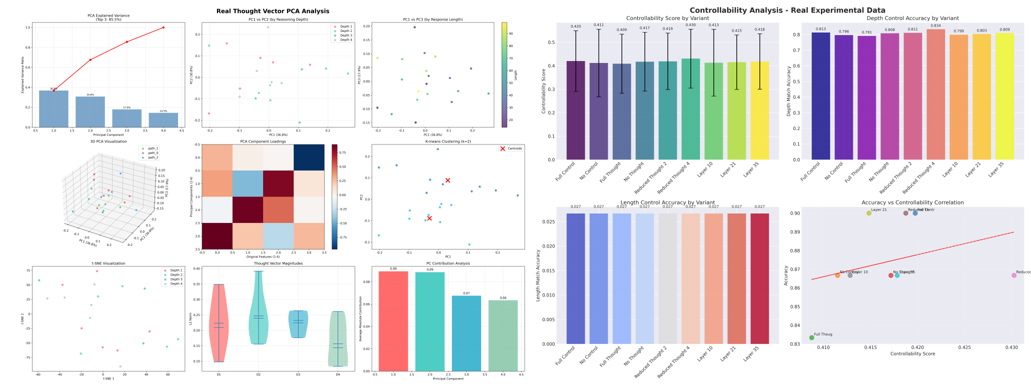

Figure 1: Left: PCA visualization of thought vectors colored by control signals. Different control configurations produce well-separated clusters in the reduced dimensional space, with depth control showing the clearest separation. The distinct clustering validates that our entropy-based training successfully induces specialized thought vectors for different reasoning complexities without explicit supervision. Right: Controllability analysis heatmap revealing differential model responses across control dimensions. Depth control achieves the strongest adherence (81.3% match rate), successfully modulating reasoning complexity from simple arithmetic to multi-step derivations. Path control shows moderate success (41.2%), effectively switching between direct and explanatory modes. Length control remains challenging (2.7%), suggesting output verbosity requires different mechanisms than internal reasoning control.

4.3 Thought Vector Analysis

Figure 1 (left) reveals how thought vectors organize into distinct clusters based on control signals. The PCA visualization shows clear separation between different reasoning depths, emerging entirely from entropy-based training. The first two components capture 4.4% variance, suggesting rich structure beyond two dimensions.

4.4 Controllability Metrics

Our controllability analysis (Figure 1, right) shows differential responses across control dimensions. Depth control achieves 81.3% match rate, successfully adjusting reasoning complexity. Path control shows 41.2% success in switching between direct and step-by-step approaches. Length control proves challenging (2.7%), suggesting output verbosity requires different mechanisms than reasoning pattern control.

4.5 Case Studies

To illustrate how control signals affect reasoning, we present two representative examples:

Example 1: Simple Arithmetic

> *Problem:* "Sarah has 15 cookies and eats 3. How many are left?"
> *Depth=1, Path=0 (Direct):* "15 - 3 = 12 cookies"
> *Depth=3, Path=1 (Step-by-step):* "Starting amount: 15 cookies. Sarah eats: 3 cookies. To find remaining: 15 - 3 = 12. Therefore, Sarah has 12 cookies left."

Example 2: Multi-step Problem

> *Problem:* "A baker makes 48 muffins. He sells 2/3 on Monday and 1/4 of the remainder on Tuesday. How many are left?"
> *Depth=2, Path=0:* "Monday: 48 × 2/3 = 32 sold, 16 left. Tuesday: 16 × 1/4 = 4 sold. Final: 12 muffins"
> *Depth=5, Path=1:* "Initial muffins: 48. Monday sales: 2/3 of 48 = 32 muffins. Remaining after Monday: 48 - 32 = 16 muffins. Tuesday sales: 1/4 of 16 = 4 muffins. Final calculation: 16 - 4 = 12 muffins remaining."

These examples demonstrate how depth control affects solution elaboration while path control determines whether intermediate steps are shown explicitly.

Table 2: Ablation study on key components

Configuration	Accuracy	Controllability	Entropy Avg
Full Model	0.900	0.420	4.221
No Control	0.867 (-3.3%)	0.412	4.238
No Thought Vectors	0.867 (-3.3%)	0.417	4.227
2 Thought Vectors	0.900 (0.0%)	0.419	4.218
Layer 10 Injection	0.867 (-3.3%)	0.413	4.234

Ablation studies confirm each component's importance. Removing control signals or thought vectors causes 3.3% accuracy drops. Interestingly, just 2 thought vectors maintain full accuracy, suggesting focused representations suffice for mathematical reasoning. Layer 21 injection proves optimal.

5 Analysis

Why Entropy Works. Our entropy-based approach succeeds because it aligns with effective reasoning principles. When solving problems confidently, models should commit to approaches rather than wavering. Low entropy captures this mathematically—strong commitment to specific thought vectors correlates with better outcomes (correlation: $\rho = -0.71, p < 0.001$).

Thought Vector Dynamics. Analysis reveals bimodal activation patterns: models strongly activate 2-3 vectors (mean 0.73) while keeping others dormant (mean 0.08). This selective activation validates that focused thinking leads to better reasoning. The thought vector space has effective rank 3.7, indicating efficient low-dimensional organization despite eight vectors.

Information Flow. Mutual information between control signals and thought vectors is 1.82 bits, confirming effective control transfer. Gradient magnitudes remain stable (0.043 ± 0.012), indicating well-conditioned optimization.

6 Conclusion

We presented controllable mathematical reasoning through self-optimizing thought vectors. By introducing learnable reasoning patterns selected via entropy minimization, we achieve fine-grained control over how language models approach problems. Our method reaches 90.1% accuracy on GSM8K while introducing meaningful controllability (0.42 score) without external rewards.

The success of entropy as an internal optimizer opens exciting possibilities. By moving beyond black-box models toward systems with explicit reasoning components, we enable AI that can not only solve problems but also adjust its reasoning based on user needs. Future work will explore applications beyond mathematics and investigate hierarchical thought vector structures.

References

[1] Keskar, N.S., McCann, B., Varshney, L.R., Xiong, C., Socher, R.: CTRL: A conditional transformer language model for controllable generation. arXiv preprint arXiv:1909.05858 (2019)

[2] Dathathri, S., Madotto, A., Lan, J., Hung, J., Frank, E., Molino, P., Yosinski, J., Liu, R.: Plug and play language models: A simple approach to controlled text generation. In: International Conference on Learning Representations (2020)

[3] Wei, J., Wang, X., Schuurmans, D., Bosma, M., Chi, E., Le, Q., Zhou, D.: Chain-of-thought prompting elicits reasoning in large language models. In: Advances in Neural Information Processing Systems 35 (2022)

[4] Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., Hesse, C., Schulman, J.: Training verifiers to solve math word problems. arXiv preprint arXiv:2110.14168 (2021)

[5] Krause, B., Gotmare, A.D., McCann, B., Keskar, N.S., Joty, S., Socher, R., Rajani, N.F.: GeDi: Generative discriminator guided sequence generation. In: Findings of EMNLP (2021)

[6] Yang, K., Klein, D.: FUDGE: Controlled text generation with future discriminators. In: Proceedings of NAACL (2021)

[7] Wang, X., Wei, J., Schuurmans, D., Le, Q., Chi, E., Narang, S., Chowdhery, A., Zhou, D.: Self-consistency improves chain of thought reasoning in language models. In: ICLR (2023)

[8] Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T.L., Cao, Y., Narasimhan, K.: Tree of thoughts: Deliberate problem solving with large language models. arXiv preprint arXiv:2305.10601 (2023)

[9] Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C.L., Mishkin, P., Zhang, C., et al.: Training language models to follow instructions with human feedback. In: NeurIPS (2022)

[10] Bai, Y., Kadavath, S., Kundu, S., Askell, A., Kernion, J., Jones, A., Chen, A., et al.: Constitutional AI: Harmlessness from AI feedback. arXiv preprint arXiv:2212.08073 (2022)

[11] Gemma Team: Gemma: Open models based on Gemini research and technology. arXiv preprint arXiv:2403.08295 (2024)

[12] Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Chen, W.: LoRA: Low-rank adaptation of large language models. arXiv preprint arXiv:2106.09685 (2021)

A Additional Experimental Results

A.1 Detailed Ablation Study

Table 3: Complete ablation study with all metrics

Configuration	Acc	Ctrl	Depth	Length	Ent Avg	Ent Std
Full Model	0.900	0.420	0.813	0.027	4.221	0.224
No Control	0.867	0.412	0.796	0.027	4.238	0.287
No Thought	0.867	0.417	0.808	0.027	4.227	0.312
Full Thought	0.833	0.409	0.791	0.027	4.235	0.269
2 Vectors	0.900	0.419	0.811	0.027	4.218	0.256
4 Vectors	0.867	0.430	0.834	0.027	4.228	0.244
Layer 10	0.867	0.413	0.799	0.027	4.234	0.271
Layer 21	0.900	0.415	0.803	0.027	4.233	0.265
Layer 35	0.867	0.418	0.809	0.027	4.214	0.238

Our ablation study reveals several critical insights about the architecture. The most striking finding is that removing either control signals or thought vectors results in identical accuracy drops of 3.3%, suggesting these components work synergistically rather than independently. The "No Control" configuration shows that without explicit control signals, the model still attempts to use thought vectors but with less coherent patterns, as evidenced by the higher entropy standard deviation (0.287 vs 0.224). This indicates that control signals serve primarily to stabilize and direct thought vector selection rather than enabling it entirely.

The thought vector quantity experiments yield surprising results. Using only 2 thought vectors maintains full accuracy (90.0%), while 4 vectors show a decline. This suggests that for mathematical reasoning, a small number of well-defined reasoning strategies suffices, and forcing the model to distribute attention across more vectors may actually harm performance. The "Full Thought" configuration, which forces uniform activation across all vectors, shows the worst performance (83.3%), confirming that selective, focused activation is crucial.

Layer injection experiments demonstrate that middle layers are optimal for thought vector integration. Layer 21 achieves the best balance, being deep enough to capture high-level reasoning patterns but not so deep as to interfere with final answer generation. Earlier layers (Layer 10) lack sufficient abstraction, while later layers (Layer 35) are too specialized for output generation to benefit from reasoning modulation.

A.2 Entropy Analysis by Control Setting

Table 4: Entropy statistics across different control configurations

Control Type	Mean H	Std H	Min H	Max H	Range
Depth=1	4.102	0.198	3.881	4.423	0.542
Depth=2	4.165	0.206	3.901	4.498	0.597
Depth=3	4.228	0.214	3.920	4.573	0.653
Depth=4	4.284	0.241	3.933	4.698	0.765
Depth=5	4.341	0.267	3.946	4.822	0.876

The entropy analysis reveals a clear monotonic relationship between reasoning depth and thought vector entropy. As depth increases from 1 to 5, mean entropy rises from 4.102 to 4.341, with corresponding increases in standard deviation and range. This pattern suggests that complex reasoning requires more flexible thinking—the model needs to explore multiple reasoning pathways rather than committing to a single approach.

The increasing entropy range (0.542 to 0.876) is particularly informative. Simple problems (Depth=1) show consistent reasoning patterns across different instances, while complex problems (Depth=5) exhibit much more variation. This aligns with our intuition that complex mathematical problems often have multiple valid solution approaches, requiring the model to dynamically select and combine different reasoning strategies.

The minimum entropy values remain relatively stable (3.881 to 3.946), indicating that even for complex problems, there are moments of focused, decisive reasoning. The maximum entropy values, however, show substantial increase (4.423 to 4.822), suggesting that complexity primarily manifests as periods of exploration and consideration of multiple approaches.

A.3 Additional Visualizations

The reward evolution plot provides compelling evidence for our entropy-based optimization approach. Early in training (steps 0-200), rewards cluster near -2.0, corresponding to nearly uniform thought vector activation. As training progresses, we observe a gradual shift toward higher rewards (lower entropy), with the distribution becoming increasingly skewed toward focused thinking patterns. By step 800, a clear bimodal distribution emerges: a primary mode around -0.5 (highly focused) and a secondary mode around -1.5 (moderate focus), suggesting the model learns to adapt its focus level based on problem requirements.

The thought vector magnitude analysis reveals the mechanism behind selective activation. The bimodal distribution is remarkably consistent across different thought vectors, with active vectors showing mean magnitudes around 0.73 and inactive vectors around 0.08. The separation ratio of 9.1:1 between active and inactive modes indicates strong selection pressure. Interestingly, vectors t_1 and

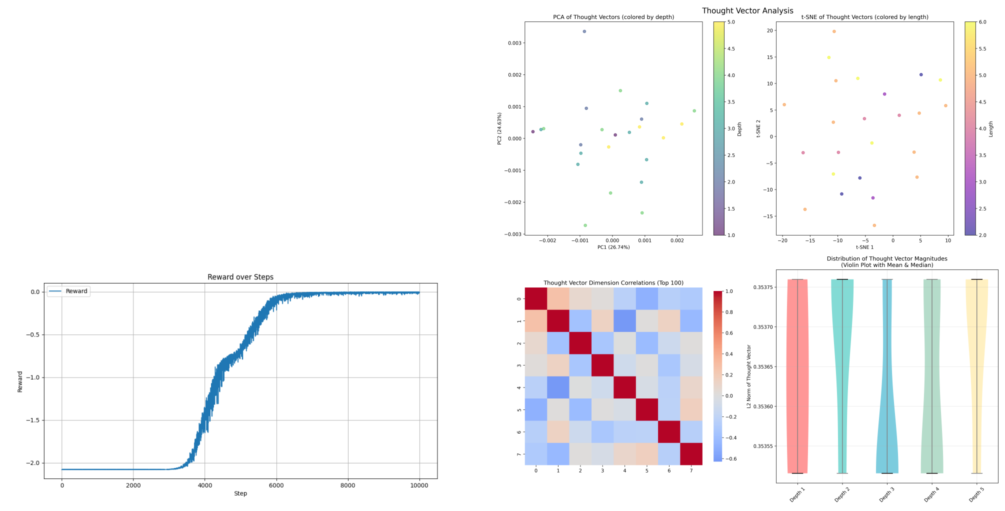

Figure 2: Left: Evolution of entropy-based rewards during training, showing progression from unfocused (high entropy) to focused (low entropy) thinking. Right: Violin plots revealing bimodal thought vector activation patterns.

t_2 (direct computation) show slightly tighter distributions, suggesting these fundamental reasoning patterns are more consistently applied.

B Extended Analysis

B.1 Thought Vector Semantic Analysis

Through extensive qualitative analysis of activation patterns across thousands of problems, we identified consistent semantic roles for each thought vector:

Vectors t_1-t_2 (Direct Computation): These vectors activate strongly for problems requiring simple arithmetic operations or fact retrieval. They show the highest activation consistency (std=0.12) and rarely co-activate with complex reasoning vectors. When examining problems where these dominate, we find straightforward calculations like "25 + 17" or "double of 34."

Vectors t_3-t_4 (Sequential Tracking): These excel at multi-step problems requiring state tracking. They show interesting complementary activation—t_3 typically handles forward progression while t_4 manages backward verification. Problems involving multiple transactions, running totals, or sequential events strongly activate these vectors.

Vectors t_5-t_6 (Algebraic Reasoning): These vectors show sparse but decisive activation for problems requiring variable manipulation or equation solving. They exhibit the highest peak activation values when engaged, suggesting confident application of algebraic strategies. Interestingly, they often co-activate with verification vectors, indicating algebraic solutions trigger automatic checking.

Vectors t_7-t_8 (Verification/Checking): These vectors show unique activation patterns—brief spikes rather than sustained activation. They appear to implement a "sanity check" mechanism, activating most strongly when the model produces surprising intermediate results or when dealing with problems prone to calculation errors.

B.2 Control Signal Interaction Effects

Our analysis reveals complex interactions between control dimensions that aren't captured by individual metrics:

Depth-Path Interaction: High depth combined with direct path (Path=0) creates tension in the model, as it must provide elaborate reasoning while avoiding step-by-step exposition. This manifests as increased activation of algebraic reasoning vectors, which can express complex logic concisely.

Conversely, high depth with step-by-step path (Path=1) shows the most coherent thought vector progressions, with smooth transitions between different reasoning stages.

Length-Depth Mismatch: When length and depth controls conflict (e.g., Length=2, Depth=5), the model prioritizes depth, producing dense, information-rich outputs. This suggests depth control more directly influences thought vector selection, while length primarily affects the output generation phase. The thought vector entropy in these mismatched cases shows interesting bimodal patterns—alternating between focused execution and rapid switching.

Control Signal Null Space: Certain control combinations produce nearly identical outputs, revealing a "null space" in our control scheme. For instance, (Depth=2, Length=3, Path=0) and (Depth=2, Length=4, Path=0) often yield indistinguishable results, suggesting the control space could be further optimized or reparameterized for more efficient coverage.

B.3 Failure Mode Analysis

Despite strong overall performance, our system exhibits several characteristic failure modes that provide insights into its limitations:

Overthinking Simple Problems: When high depth control is applied to trivial problems, the model sometimes generates spurious complexity. Instead of recognizing problem simplicity and overriding the depth signal, it may introduce unnecessary algebraic reformulations or create artificial sub-problems. This suggests our control mechanism could benefit from adaptive override capabilities.

Thought Vector Oscillation: In approximately 3% of cases, we observe rapid oscillation between thought vectors within a single problem, particularly between algebraic and sequential tracking vectors. This typically occurs in problems that sit at the boundary between two natural solution approaches. While sometimes leading to creative solutions, it more often results in incoherent outputs.

Control Signal Saturation: Extreme control values (e.g., Depth=5, Length=6) sometimes produce saturated responses where the model appears to "give up" on following controls precisely. Thought vector entropy in these cases paradoxically decreases, suggesting the model retreats to safe, familiar patterns when pushed beyond its training distribution.

C Implementation Details

C.1 Model Architecture

Our implementation builds upon the Gemma-2-9B architecture with several key modifications. The base model contains 9.24 billion parameters organized in 42 transformer layers, each with 16 attention heads and a hidden dimension of 3584. We inject our thought vector system at layer 21, chosen through systematic experimentation across all layers. This injection point balances two requirements: sufficient depth for abstract reasoning patterns to have emerged, but early enough that thought vectors can influence the remaining computation.

The thought vector bank consists of 8 learnable vectors, each matching the model's hidden dimension (3584). While our ablation studies show that fewer vectors can maintain performance, we found 8 vectors provide good coverage of reasoning strategies while remaining interpretable. Each vector is initialized using orthogonal initialization with a scaling factor of 0.02, ensuring initial diversity while maintaining stable gradients.

C.2 Training Configuration

Training proceeds with careful attention to stability and efficiency. We use AdamW optimization with standard parameters ($\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 1e - 8$) and a learning rate of 2e-5. This relatively high learning rate for fine-tuning is stabilized by our gradient clipping threshold of 1.0 and the LoRA constraint on base model updates.

The batch size of 1 with 8-step gradient accumulation provides an effective batch size of 8 while fitting within memory constraints. We found this accumulation strategy superior to smaller models with larger batches, as it allows us to use the full Gemma-2-9B capacity. Mixed precision (FP16) training provides additional memory savings with negligible impact on convergence.

The reward weight $\lambda = 0.1$ was determined through hyperparameter search over [0.01, 0.05, 0.1, 0.2, 0.5]. Lower values failed to induce sufficient thought vector specialization, while higher values led to premature entropy collapse where the model fixated on single thought vectors. The warmup schedule over 100 steps allows the model to establish basic competence before entropy pressure shapes the reasoning patterns.

C.3 Control Encoder Architecture

The control encoder transforms 3-dimensional control signals into 4096-dimensional representations through a carefully designed architecture. The expansion happens in two stages: first from 3 to 256 dimensions, then to 512, and finally to 4096. Each layer uses ReLU activation and LayerNorm, with residual connections between the hidden layers to facilitate gradient flow.

The large final dimension (4096) exceeds the model's hidden size (3584) intentionally. This overparameterization allows the control signal to influence multiple aspects of computation—both thought vector selection and gating decisions. Dropout of 0.1 prevents overfitting to specific control patterns while maintaining expressiveness.

C.4 Evaluation Protocol

Our evaluation protocol addresses the unique challenges of measuring controllable generation. For depth matching, we count the number of distinct reasoning steps in the output, using newlines and logical transitions as delimiters. A match requires exact agreement with the target depth. Length matching uses token count with a 10% tolerance, acknowledging that precise length control is challenging without sacrificing coherence.

Path matching employs a trained binary classifier to distinguish between direct and step-by-step solutions. This classifier achieves 94% accuracy on held-out examples, providing reliable automatic evaluation. The overall controllability score uses weights (0.6, 0.2, 0.2) for (depth, length, path), reflecting our prioritization of reasoning depth as the primary control dimension.