Fuel Gauge: Estimating Chain-of-Thought Length Ahead of Time in Large Multimodal Models

Appendix

A. More Details on the Experimental Setup

We perform all experiments on a dedicated server equipped with an AMD EPYC 9554 CPU and eight NVIDIA RTX A6000 GPUs, each with 48 GB of VRAM. Although the machine hosts multiple GPUs, all training and inference experiments for this paper are executed using a single GPU, ensuring reproducibility and eliminating cross-GPU variability. The system runs a standard Linux environment with CUDA-compatible drivers suitable for large-scale model training.

Our implementation uses PyTorch 2.8.0 [46] as the primary deep learning framework. For text-only experiments, we rely on HuggingFace Transformers v4.53.2 [45]; this provides stable support for the language models evaluated in these tasks. For multimodal and other non–text-only settings, we use Transformers v4.57.0, as this version includes updated support for the vision-language models and additional architectural features required by our experiments.

Training Procedure for f_{fuel} and f_{sig} Section 4.4 provides a high-level overview of the training strategy for the fuel estimator f_{fuel} and the signal extractor f_{sig}. Here, we elaborate on the full training configuration. Both modules are trained independently using the AdamW optimizer [47] with a base learning rate of 10^{-3}, and a weight decay coefficient of 10^{-4}. To ensure stable optimization during the early phase of training, we employ a cosine learning-rate schedule with 1000 warm-up steps [48], allowing the model to gradually adapt before entering the main decay phase.

We train each model for a single epoch, which is sufficient due to the dense supervision signal and the relatively low-dimensional nature of both predictors. During training, batches are shuffled to improve statistical efficiency, and gradient updates are applied without gradient accumulation on our single-GPU setup. After the training epoch completes, we evaluate the models on a held-out validation set and save the resulting checkpoints for all subsequent inference-time experiments.

B. Additional Experimental Results

B.1. Experimental Results on LMM Intern-S1 and GLM-4.6V

In this section, we expand our evaluation to Intern-S1-mini-8B [3] and GLM-4.6V [31] model. Table S5 presents the experimental results on CoT length estimation (Section 6.3). We also introduce a new baseline "LSTM" which predicts the CoT length with a recurrent LSTM model. Figure S7 shows the results for CoT length modulation (Section 6.5). Accross all three tasks shown in the Table S5, our Fuel Gauge achieves significantly better results than baseline methods. Moreover, as shown in Figure S7, our Fuel Gauge provides highly effective and stable control over the LMM CoT process: a single factor η allows linear adjustment of CoT length, which in turn translates linearly to model accuracy. In summary, these additional results on the Intern-S1-mini-8B and GLM-4.6V model further demonstrate the wide-applicability of our method.

Method	Cross-Task ↓		Cross-Modality Cross-Task ↓			
	MathVision-m		LVB-15		LVB-60	
	Intern	GLM	Intern	GLM	Intern	GLM
Mean	0.8550	0.7028	0.8598	0.8271	1.130	0.8655
Direct	0.8715	0.5187	6.441	2.507	9.733	2.805
LSTM	0.5799	0.7764	0.7553	0.8766	1.154	1.102
FuelGauge	**0.4194**	**0.3561**	**0.4902**	**0.3896**	**0.4525**	**0.4032**

Table S5. CoT length prediction evaluation results for Intern-S1-mini-8B (Intern) and GLM-4.6V (GLM) on MathVision-m, LongVideoBench-15 (LVB-15), and LongVideoBench-60 (LVB-60) with an additional baseline "LSTM". MathVision-m demonstrates the generalizability of Fuel Gauge from general tasks to specialized tasks, while LongVideoBench highlights its generalizability across both tasks and modalities (*i.e.*, from text-image to text-video). Across all three benchmarks, Fuel Gauge significantly outperforms the baseline methods.

B.2. Example of Predictive KV Cache Allocator

Figure S8 presents five examples of predictive KV-cache allocation (Section 5.1). A standard KV-cache allocator cannot anticipate future memory needs and therefore allocates memory only when necessary, resulting in many small allocations—visualized as a diagonal line in Figure S8. In contrast, our Fuel Gauge allows the allocator to predict memory requirements in advance, enabling it to allocate a large block of memory in a single step, shown as the "stairs" pattern in Figure S8. As demonstrated in Table 3 and Figure S8, this approach greatly reduces the frequency of memory allocation and release, thereby mitigating memory fragmentation.

B.3. Detailed Results for CoT Length Modulation in Figure 6

Figures S11, S12, S13, and S14 present more detailed results on CoT modulation (placed at the end of the Appendix due to the spacing issues). Across all combinations of LMMs and benchmarks, our Fuel Gauge method consistently enables effective, linear control over CoT length,

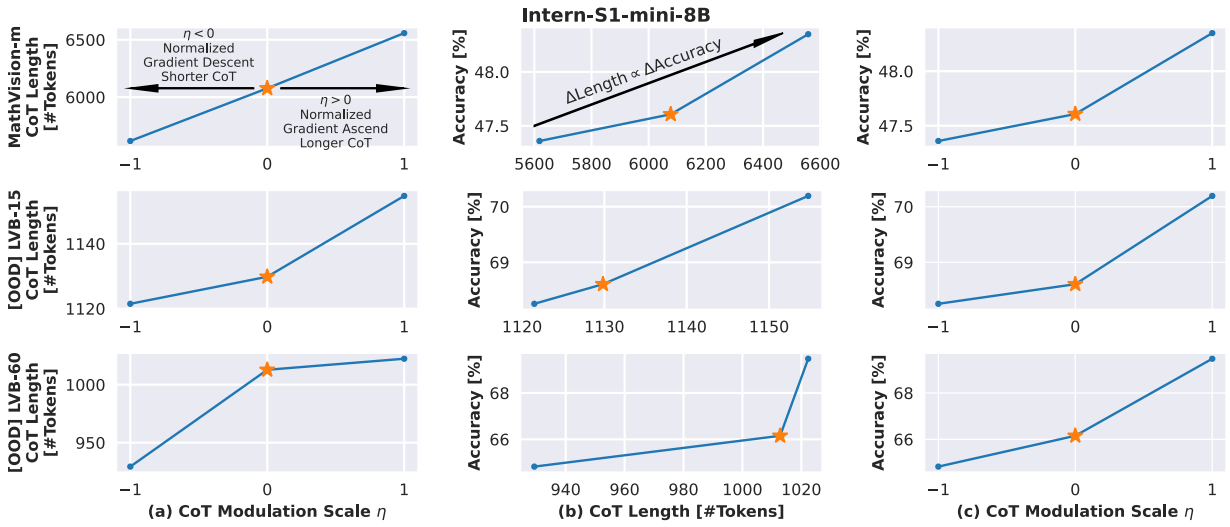

Figure S7. CoT length and LMM accuracy for Intern-S1-mini-8B with different CoT modulation factors η. Orange star denotes the baseline case where no CoT modulation is applied. Figure (a) shows that Fuel Gauge controls the CoT length linearly. Figure (b) shows that the change in CoT length quasi-linearly translates to a change in accuracy. Finally, Figure (c) shows that based on the linearity in figures (a) and (b), we can achieve our target and control the accuracy quasi-linearly with η.

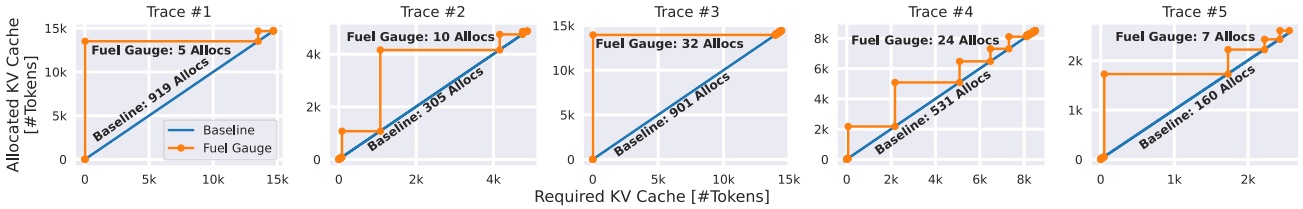

Figure S8. Examples of predictive KV-cache allocation. The baseline allocator (HF, Section 6.1) performs frequent small, on-demand allocations (diagonal line in plots). With our Fuel Gauge, memory needs are predicted and allocated in larger chunks (stair line in plots), reducing allocation frequency and mitigating fragmentation.

which in turn leads to correspondingly linear changes in accuracy. This high level of controllability allows users to efficiently intervene when an LMM is either under- or overthinking.

These results also provide additional support for the hypotheses in Sections 4.1 and 4.2. If either hypothesis were incorrect, the Fuel Gauge would likely fail to maintain stable and consistent linear control of CoT length (see the rationale at the beginning of Section 6). The observed linear relationships therefore reinforce both hypotheses and confirm the effectiveness of our Fuel Gauge.

C. Ablation Study

C.1. Number of Parameters for f_{sig} and f_{fuel}

We change the number of parameters for f_{sig} and f_{fuel} by varying the number of channels for the convolutional layers and MLP layers of the model.

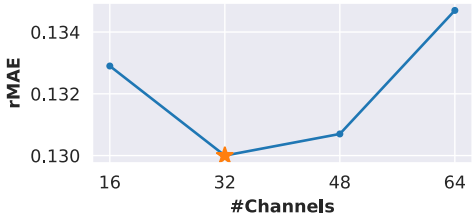

Figure S9. Fuel level estimation accuracy with different number of channels for f_{sig} and f_{fuel} for Qwen3-4B on MMLU. Star denotes the adopted design choice for Fuel Gauge.

As shown in Figure S9, our design choice of 32-channels in f_{fuel} and f_{sig} yields the best fuel level estimation accuracy. A channel number other than 32 leads to either overfitting

Loss Function	Test rMAE ↓
MSE	0.1323
MAE	0.1311
Smooth L1 ($\beta = 0.1$)	0.1317
Smooth L1 ($\beta = 0.01$)	**0.1300**
Smooth L1 ($\beta = 0.001$)	0.1311

Table S6. Fuel level estimation accuracy with different loss function in Equation 3 for Qwen3-4B on MMLU. Bold denotes the adopted design choice for Fuel Gauge.

or under-fitting issue on the training samples. This supports our design choice for the Fuel Gauge.

C.2. Window Size for f_{sig}

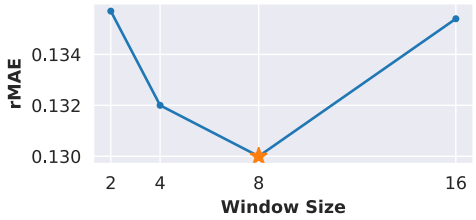

Figure S10. Fuel level estimation accuracy with different sizes of window adopted in f_{sig} for Qwen3-4B on MMLU. Star denotes the adopted design choice for the Fuel Gauge.

Figure S10 shows the evaluation result for changing the window size adopted in f_{sig}. For example, as described in Section 4.4, at time step i, we use only the most recent 8 hidden states $h_{i-7:i}$ recorded from LMM as the input to f_{sig}, namely, with a window size of 8. As shown in Figure S10, changing the window size other than 8 leads to a worse performance. This is because using a larger window size may lead to the staleness in updating prediction while using a smaller window size provides insufficient information to f_{sig} for hidden signal extraction, whereas our design choice of a window size 8 strikes the balance between agileness in updating the prediction and quality in prediction, thus supporting our design choice.

C.3. Loss Function in Equation 3

Table S6 shows the evaluation results for changing the loss functions adopted in the training process for f_{sig} and f_{fuel} in Equation 3. We consider three loss functions: mean-square error (MSE), mean absolute error (MAE) and smooth L1 loss (smooth L1), where the smooth L1 loss can be formulated as following:

$$L_{SL1}(y_{i,t}, \hat{y}_{i,t}) = \begin{cases} 0.5(y_{i,t} - \hat{y}_{i,t})^2/\beta, & |y_{i,t} - \hat{y}_{i,t}| < \beta \\ |y_{i,t} - \hat{y}_{i,t}| - 0.5\beta, & |y_{i,t} - \hat{y}_{i,t}| \geq \beta \end{cases} \tag{7}$$

In Equation 3, we adopt smooth L1 with $\beta = 0.01$ as our loss function. As shown in Table S6, our design choice yields the best performance on the benchmark. This is because the smooth L1 with $\beta = 0.01$ strikes the best balance between the MSE loss, which is sensitive to errors but not robust to outliers, and the MAE loss, which is robust to outliers but not sensitive to large errors.

Model	Throughput	#Parameters
Qwen3-4B	22.4 Token / s	4B
Fuel Gauge [Batch Size=1]	792.7 Token / s	82.24k
Fuel Gauge [Batch Size=32]	11217.3 Token / s	82.24k

Table S7. Overhead of Fuel Gauge on NVIDIA A6000 GPU. Results show that the time and memory spent in Fuel Gauge is negligible.

D. Overhead Analysis

Given its small size, the overhead of our Fuel Gauge is negligible. As shown in Table S7, the Fuel Gauge for the Qwen3-4B model contains only 82k parameters and processes 792.7 tokens per second with a batch size of 1. Increasing the batch size to 32 boosts the throughput to 11k tokens per second. In comparison, the Qwen3-4B model itself has 4 billion parameters with only 22.4 tokens per second throughput. Thus, the additional cost of the Fuel Gauge is indeed minimal.

Figures for CoT Length Modulations in Section B are in Next Page.

References

[46] Paszke, Adam, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen et al. "Pytorch: An imperative style, high-performance deep learning library." Advances in neural information processing systems 32 (2019). 1

[47] Loshchilov, Ilya, and Frank Hutter. "Decoupled weight decay regularization." arXiv preprint arXiv:1711.05101 (2017). 1

[48] Loshchilov, Ilya, and Frank Hutter. "Sgdr: Stochastic gradient descent with warm restarts." arXiv preprint arXiv:1608.03983 (2016). 1

[49] Casella, George, and Roger Berger. Statistical inference. Chapman and Hall/CRC, 2024.

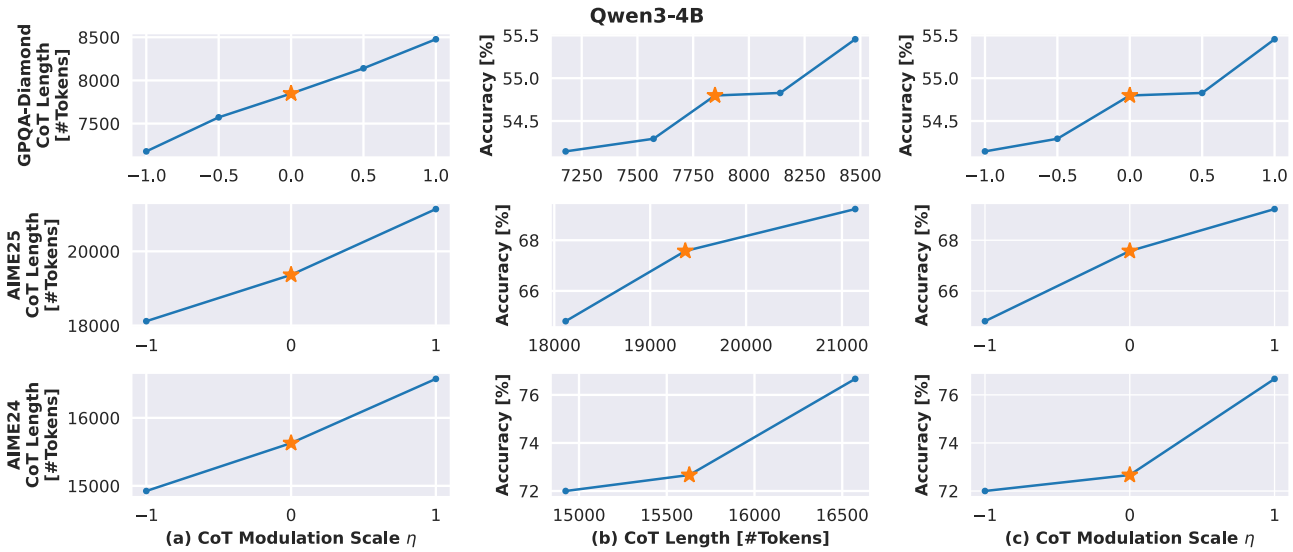

Figure S11. CoT length and LMM accuracy for different CoT modulation factors η. Results are obtained with Qwen3-4B model on multiple benchmarks. Orange star denotes the baseline case where no CoT modulation is applied. Figure (a) shows that Fuel Gauge controls the CoT length linearly. Figure (b) shows that the change in CoT length quasi-linearly translates to a change in accuracy. Finally Figure (c) shows that based on the quasi-linearity in figures (a) and (b), we can achieve our target and control the accuracy quasi-linearly with η.

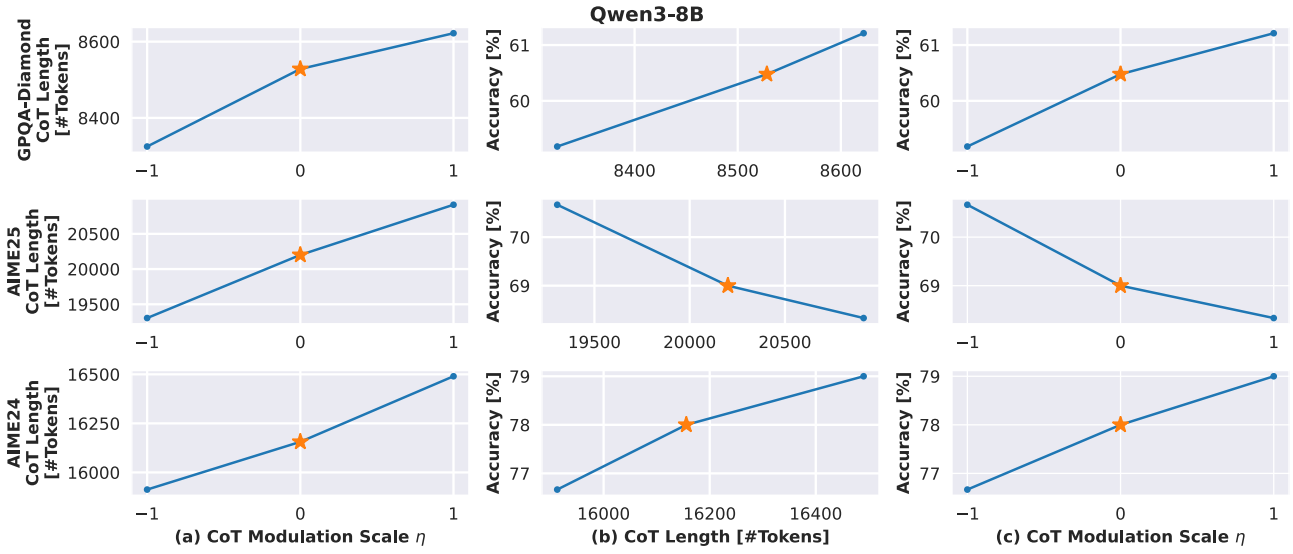

Figure S12. CoT length and LMM accuracy for different CoT modulation factors η. Results are obtained with Qwen3-8B model on multiple benchmarks. Orange star denotes the baseline case where no CoT modulation is applied. Figure (a) shows that Fuel Gauge controls the CoT length linearly. Figure (b) shows that the change in CoT length quasi-linearly translates to a change in accuracy. Finally Figure (c) shows that based on the quasi-linearity in figures (a) and (b), we can achieve our target and control the accuracy quasi-linearly with η.

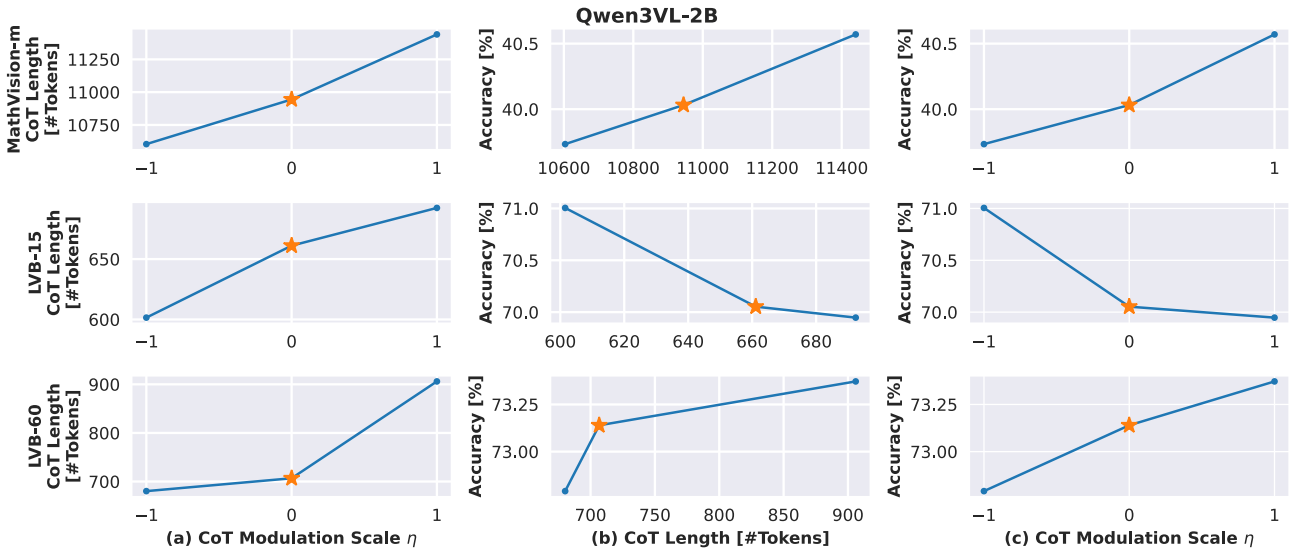

Figure S13. CoT length and LMM accuracy for different CoT modulation factors η. Results are obtained with Qwen3VL-2B model on multiple benchmarks. Orange star denotes the baseline case where no CoT modulation is applied. Figure (a) shows that Fuel Gauge controls the CoT length linearly. Figure (b) shows that the change in CoT length quasi-linearly translates to a change in accuracy. Finally Figure (c) shows that based on the quasi-linearity in figures (a) and (b), we can achieve our target and control the accuracy quasi-linearly with η.

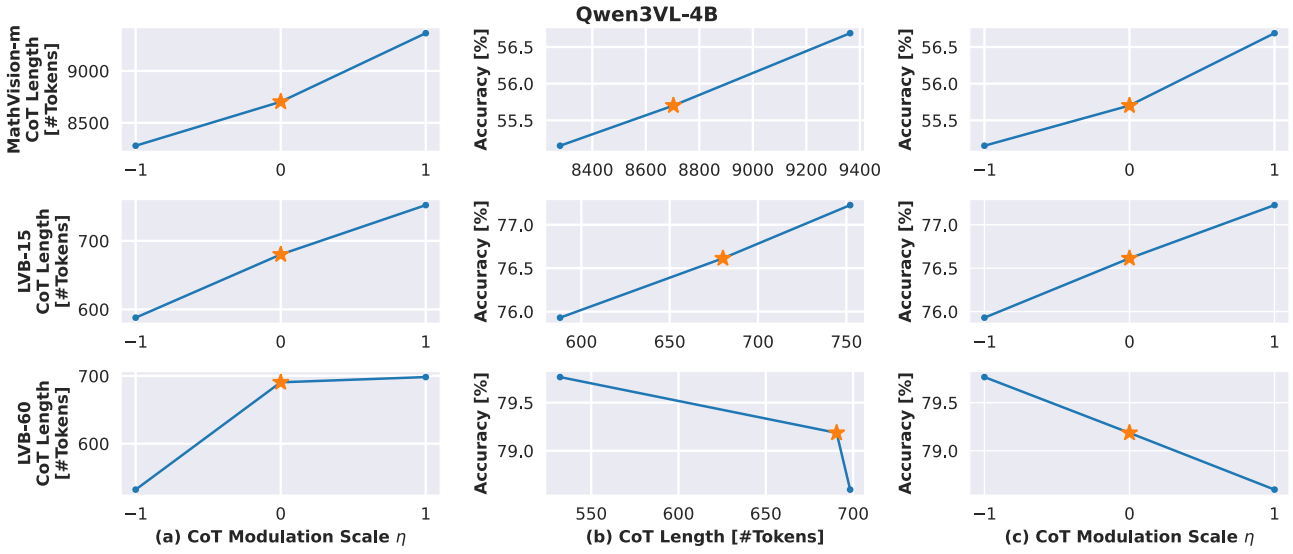

Figure S14. CoT length and LMM accuracy for different CoT modulation factors η. Results are obtained with Qwen3VL-4B model on multiple benchmarks. Orange star denotes the baseline case where no CoT modulation is applied. Figure (a) shows that Fuel Gauge controls the CoT length linearly. Figure (b) shows that the change in CoT length quasi-linearly translates to a change in accuracy. Finally Figure (c) shows that based on the quasi-linearity in figures (a) and (b), we can achieve our target and control the accuracy quasi-linearly with η.