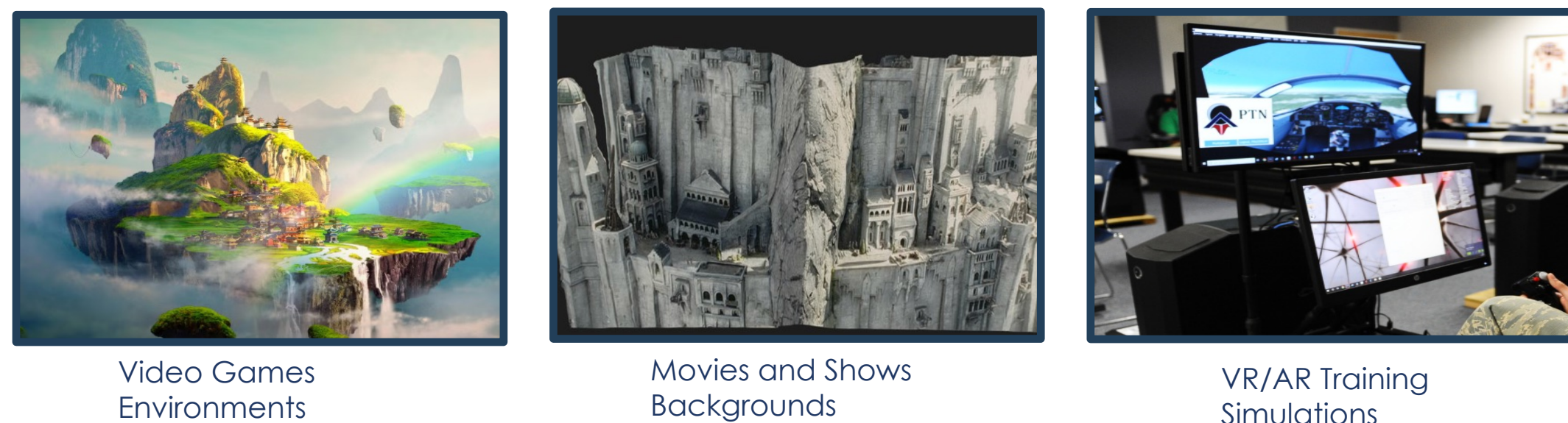
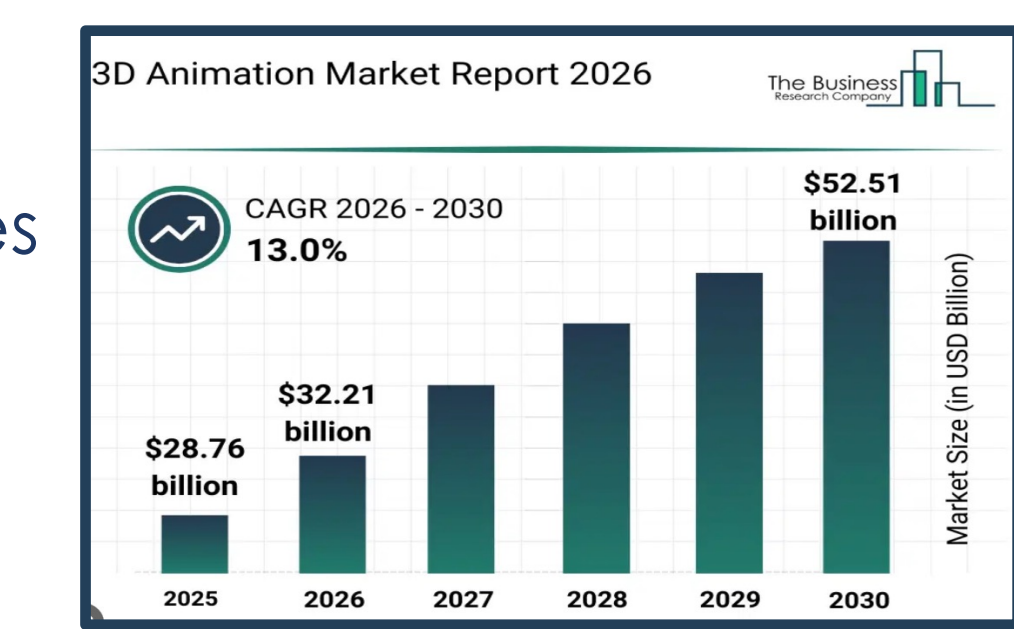


Background & Motivation

3D Scenes are Everywhere



The Production Bottleneck

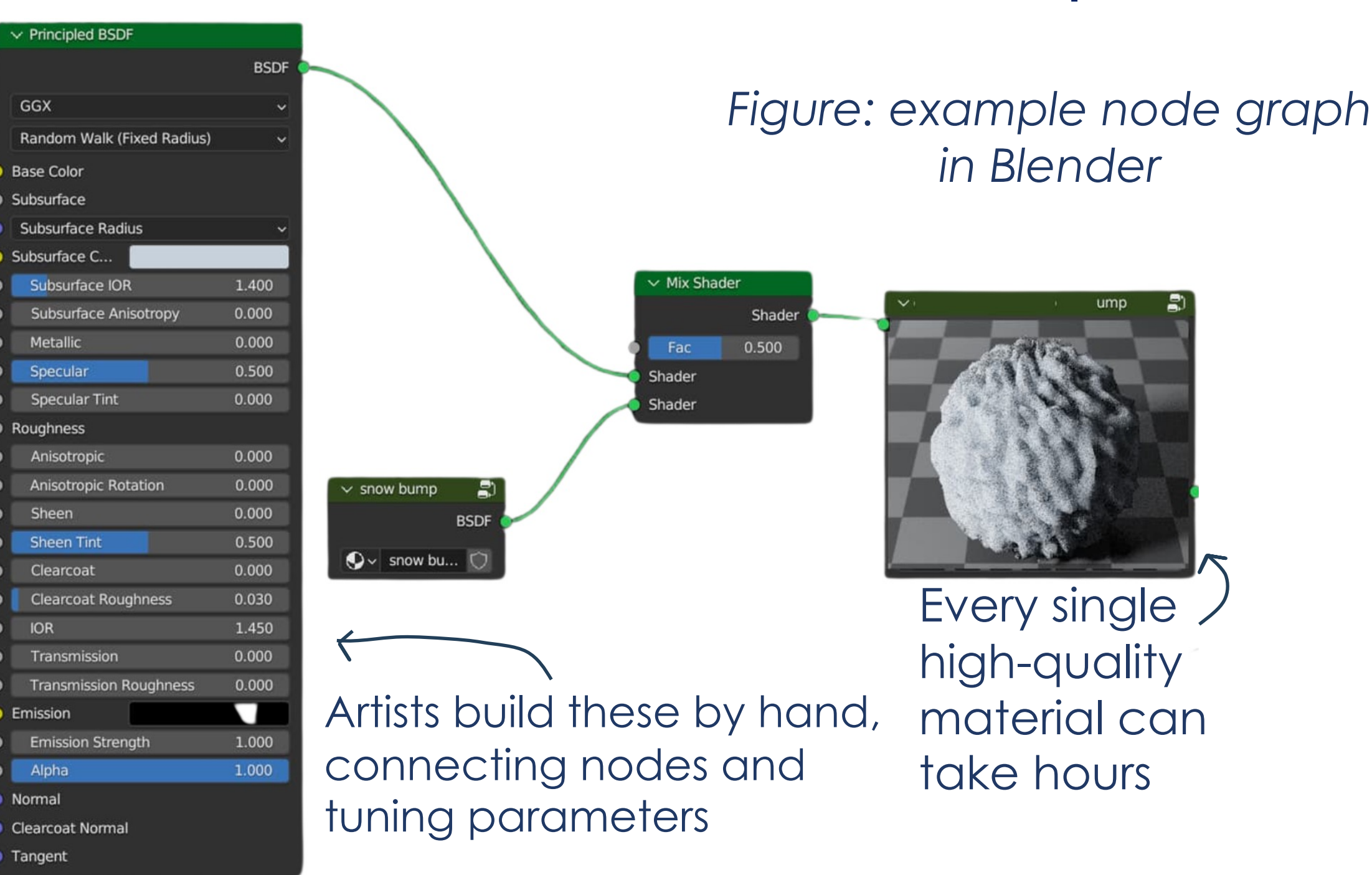


Demand is highly profitable, and the the market is projected to grow by **100 percent** in the next 5 years.

Materials Drive Realism

Even if the geometry of a scene is right, without proper materials the entire scene looks fake: **accuracy is crucial**

Blender Materials + Shader Node Graphs



Prior Image-Based Approaches

(VLMaterial, MatFormer, Hu et al.):

Train on pairs of images and code, where the image shows a material and the **code is the node graph that produces it**

These have shown that **models can learn to generate shader graphs** directly from images which we use and need to replicate

Reproduces **familiar** training-set materials well

But it's pattern matching, they copy appearance **without reasoning** about node choice or parameter values

They fail on novel, unusual or complex material beyond training

Our Approach

- Tutorials give the model things that a single image just can't:
 - Procedure** — the steps the artist actually follows
 - Parameters** — which settings matter and what ranges tend to work
 - Preferences** — the reasoning behind each design choice
- Tutorials show the *process* of making a material, not just the final result, which is what a model needs if it's going to handle materials it hasn't seen before

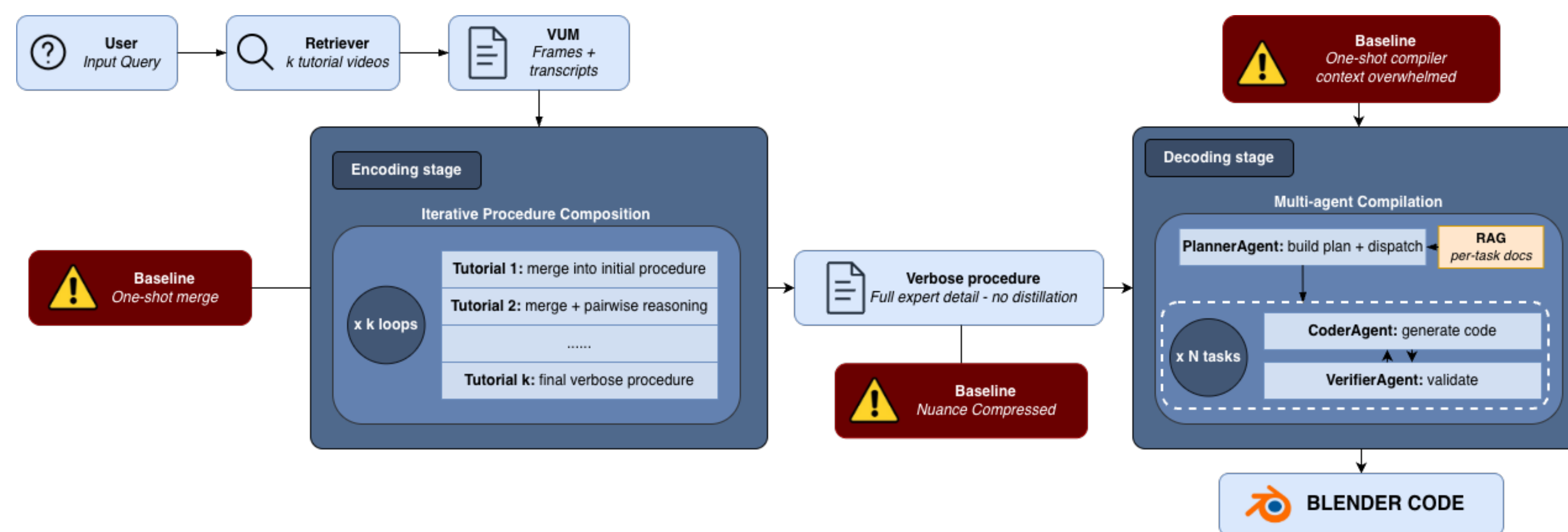
Problem Statement

Current tutorial-video pipelines lose expert detail in two places: tutorials are merged in **one shot**, and the resulting procedure is **distilled** before code generation. Both steps discard information that expert artists rely on.

We redesign both stages:

- Iterative procedure composition:** merge tutorials step by step
- Progressive compilation:** generate code in focused subtasks, each with full access to the verbose procedure

Methodology



Analysis & Challenges

Cost Tradeoff

Best average CLIP (0.847). Multi-stage reasoning outperforms one-shot generation on every material.

High Cost at Full Pipelines

Integrated costs 47% more than baseline, limits scalability without prompt compression

Diminishing returns

CLIP peaks at iteration 1 for all materials. Later iterations add noise rather than refinement.

Encoding Drivers Gain

Encoding alone lifts Steel and Pineapple but costs Mahogany 0.043.

Encoding Drives Gain

9% cheaper than baseline and slightly above it on CLIP. Best quality-per-dollar ratio of all methods.

Organic Materials

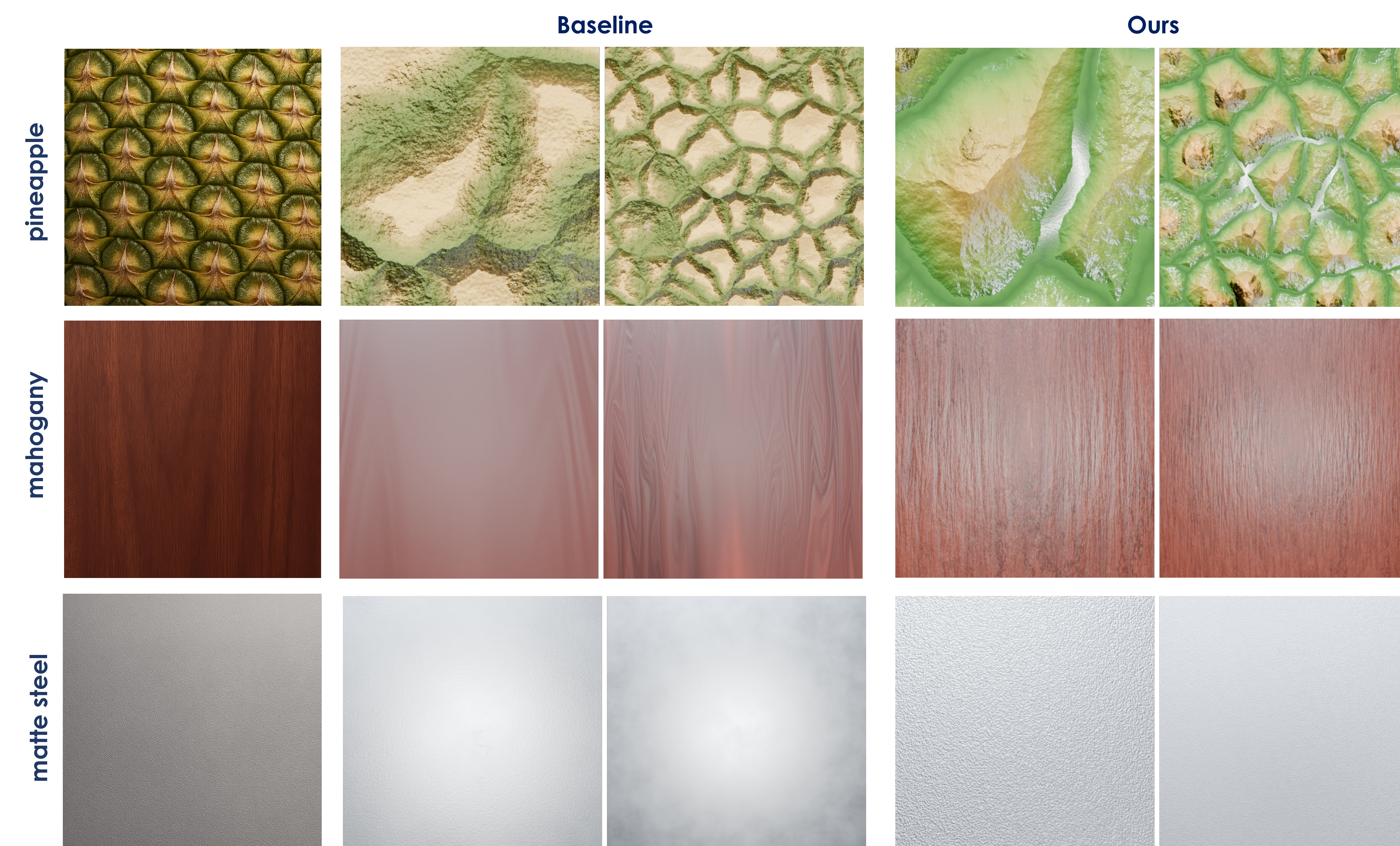
Pineapple Skin scores lowest across all methods. Irregular geometry resists procedural description.

Takeaway

Iterative encode-then-compile consistently beats one-shot generation across all materials and metrics.

Results

Original Image vs. Baseline vs. Integrated Pipeline



CLIP cosine similarity

Method	Steel	Wood	Pine	Avg
Baseline	0.859	0.859	0.775	0.831
Encoding Ony	0.892	0.816	0.810	0.839
Compilation Only	0.878	0.804	0.762	0.814
Integrated	0.894	0.862	0.784	0.847

Avg. Cost Per Run (\$)

Method	Avg. cost	Avg. CLIP	Cost Variance
Baseline	\$3.48	0.831	0.150
Encoding Ony	\$3.15	0.839	0.123
Compilation Only	\$4.54	0.814	0.389
Integrated	\$5.10	0.847	0.244

Acknowledgements & References

This work builds upon the VLMaterial framework by Beichen Li et al. and the MaterialApprentice pipeline developed by Kunal Gupta et al.