

Peter/Zheyuan HU




Homepage: <https://peterhuistyping.github.io/>

Github: <https://github.com/PeterHUistyping>

CV: https://peterhuistyping.github.io/asset/doc/CV_PeterHU.pdf

EDUCATION


 **University of Cambridge**, United Kingdom

Oct 2025-Jun 2026

M.Eng. (Hons) Computer Science.

[Jardine Scholarship](#)

Selected from top undergrads, on par with the M.Phil in research depth and assessment rigor, stated by [the department](#).

 **University of Cambridge**, United Kingdom

Oct 2022-Jul 2025

B.A. (Hons) Computer Science | First-Class (72.4) | Dissertation (93.5).

[Jardine Scholarship](#)

OS, DB, Architecture, Graphics, XR, Network, BioInfo, Quantum Computing, Information Theory, etc | [detailed notes](#).

 **Universitas Amaiensis**, **Project 985** & Top 1 in Southern China

Sep 2021-Jun 2022

B.Eng. undergrad in Software Engineering | Rank 1/173 (1st term) | Yearly score (88.2).

Withdrew after 1st year

C and C++, Object-Oriented Programming, Calculus and Linear Algebra, University Physics, Presentation, ACM, SSE.

Jardine Scholarship, the Jardine Foundation

Oct 2022-Jun 2026

merit-based, fully-funded Scholarship while pursuing my four-year studies at the University of Cambridge ([certificate](#)).

TECHNICAL SKILLS

Data Sci: Prob and Stat, Python, NumPy, ML&DL, PyTorch, Computer Vision.

Visual: Computer Graphics, OpenGL, GLSL, XR (AR/VR/MR), Unity, Unreal Engine, 3D Modelling (Blender).

Prog: C/C++, Java, OOP, CMake, gdb, Algorithms and Data Structure, OCaml (Functional Programming).

Dev Tools: bash/shell, git, CI/CD pipeline, Docker, VS Code, Pycharm, IntelliJ IDEA.

For details, please refer to my CV: https://peterhuistyping.github.io/asset/doc/CV_PeterHU.pdf

NeuMaDiff: Neural Material Synthesis via Hyperdiffusion

Chenliang Zhou, *Zheyuan Hu*, *Alejandro Sztrajman*, *Yancheng Cai*, *Yaru Liu*, *Cengiz Öztireli*.

2024 - 2025 | *Computer Graphics (real-world materials), Vision (generation via PCA, VAE, diffusion)*.

Adapted from my undergrad [dissertation project](#) (93.5, rank 1/133).



Coauthored with my supervisors *Chenliang Zhou* and Dr *Alejandro Sztrajman*, we together propose different aspects to the novel architecture, where I draw the network architecture via LaTeX *TikZ*. Our methods support the unconditional, statistically constrained and multi-modal generation of materials. With my experiments on PCA, VAE taking in either original or neural materials, we found out that hyperdiffusion has superiority in generating materials with high fidelity and diversity.

Team up with Dr *Alejandro Sztrajman*, we further explore and finalize the statistical constraints for each type of material, e.g. plastic, fabric, metallic, mirror-like, etc, based on observation and experiments. It serves as a classical, as well as ML-free way to classify the material.

In particular, there is a lack of effective BRDF-space metrics in generation tasks, where we fill the gap through my experiments, targeting the diversity and fidelity with various underlying distance functions tried. Through literature review, I raise the idea of adapting point cloud based metrics to the field of materials. With further discussions with *Chenliang Zhou*, we finalize on which pairwise distance measurements to use. To further illustrate its effectiveness, I utilize the confusion matrix to demonstrate the evaluation process. With help of my plotted graph, it's clear that our methods are able to identify the relationship between reference and synthetic material sets.

Finally, as a proof of concept, I setup some typical scenes using the proposed materials and render them with the physically-based renderer *Mitsuba3*, which I writeup the BSDF class taking in the required binary format with proper importance sampling strategies (since there's no such public error-free implementation for Python version Mitsuba3).

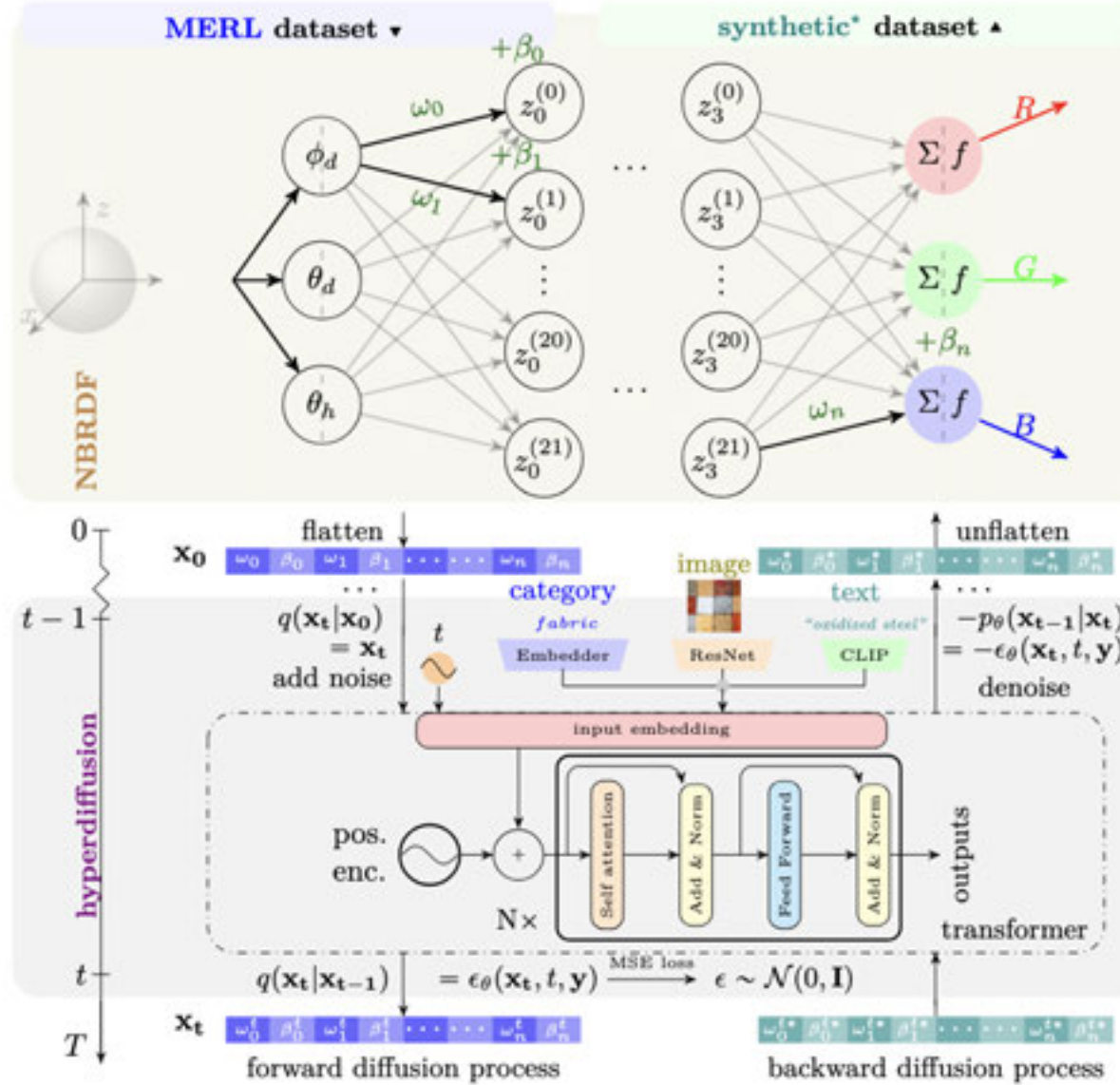


Figure 3.1: An overview of the main pipeline used in the dissertation.

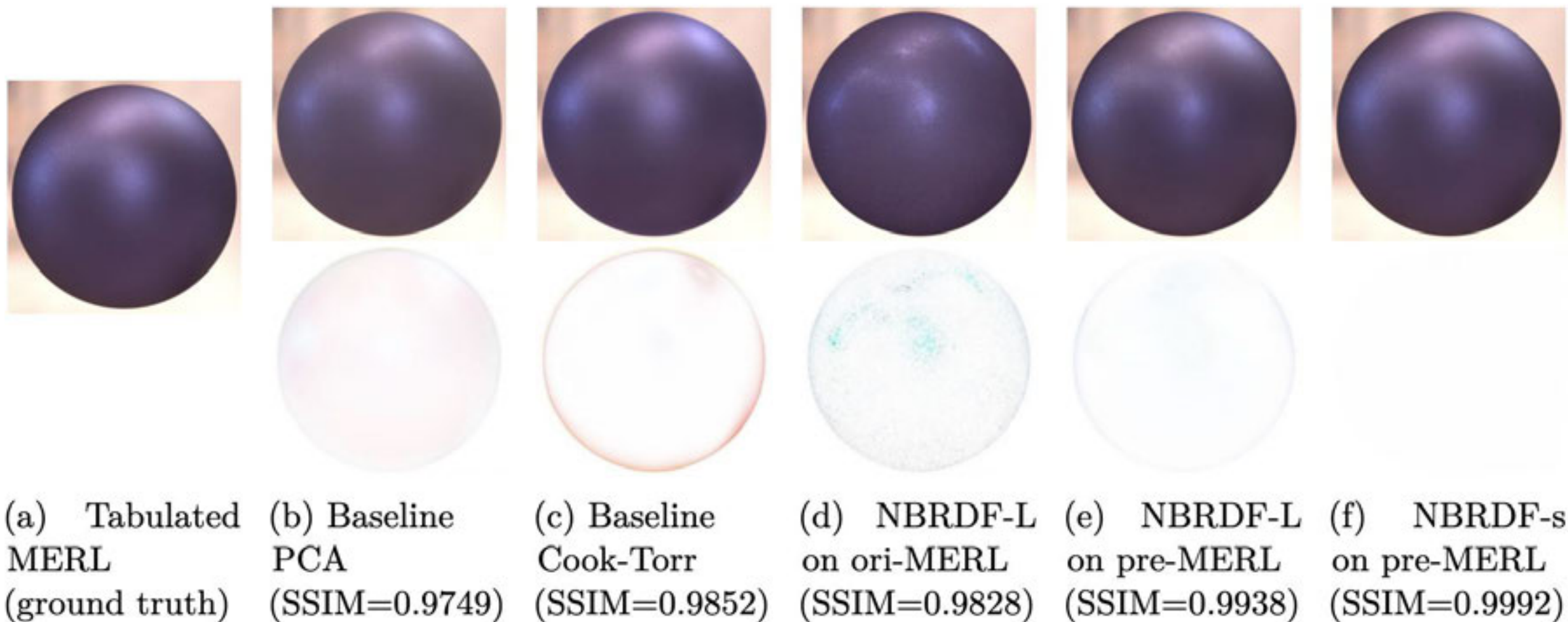
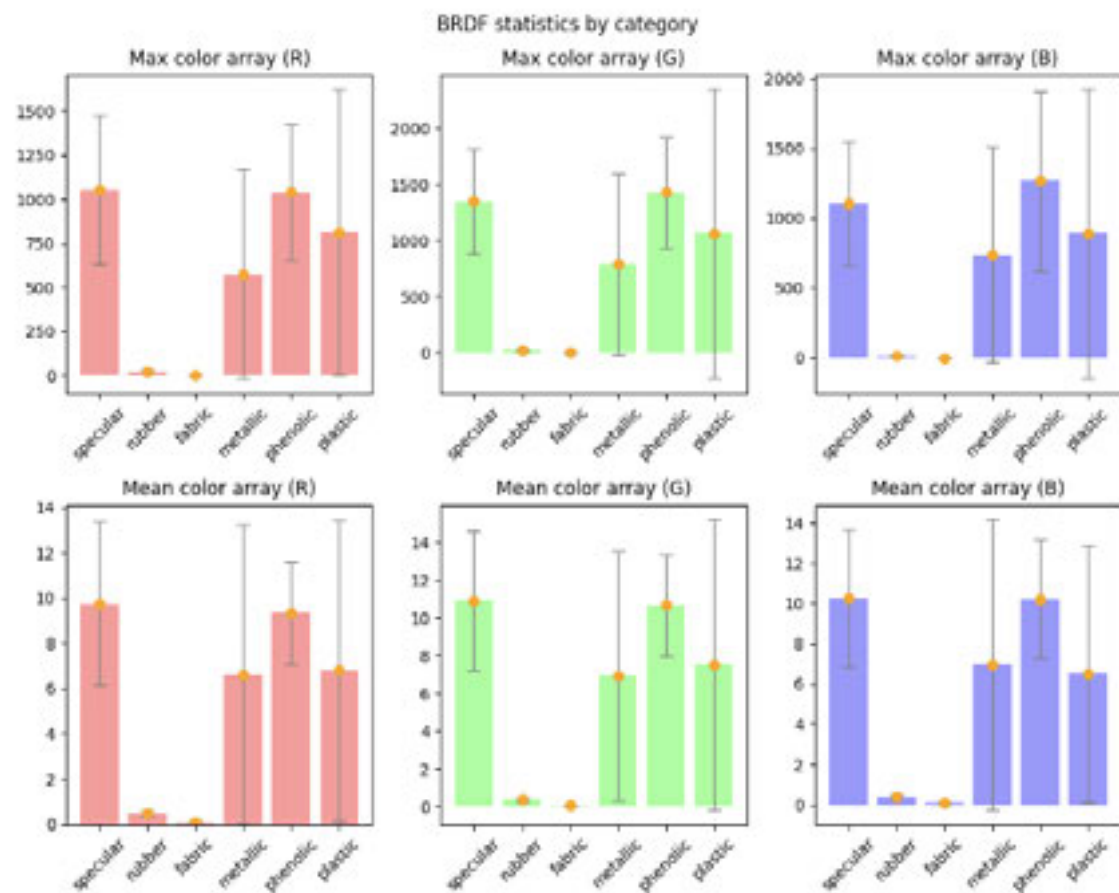


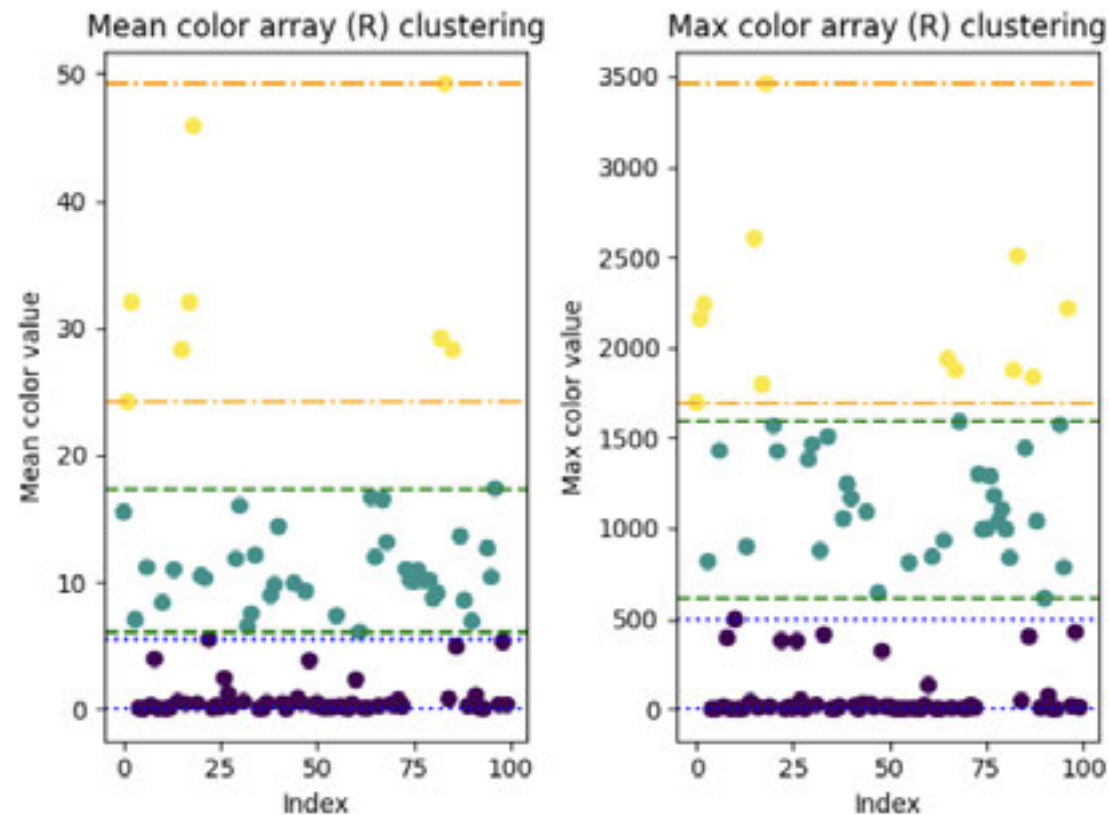
Figure 4.1: Rendered images and SSIM index maps comparison from different BRDF models against ground truth (material: blue-metallic-paint).

	metrics	training set	PCA-b	PCA-m	VAE-b	VAE-m	hd-wo-aug	hyperdiff
(←) MMD	RMSE $\times 10^2$	7.54	33.3	30.2	63.7	69.2	13.4	9.34
	-PSNR	-28.7	-13.9	-14.8	-8.30	-7.40	-22.6	-25.6
	-SSIM $\times 10$	-9.55	-6.74	-6.29	-2.68	-2.15	-8.27	-9.40
	L1 $\times 10^{-3}$	2.51	9.05	9.22	9.09	∞	4.30	4.02
	L1-log	1.05	2.72	1.66	3.70	14.6	1.16	1.21
	L1-log-cos	0.620	2.35	1.35	3.13	12.4	0.750	0.765
(→) COV %	RMSE	55.8	18.3	28.3	0.833	0.833	25.0	50.0
	-PSNR	56.7	18.3	28.3	0.833	0.833	25.0	50.0
	-SSIM	59.2	23.3	16.7	0.833	0.833	22.5	51.7
	L1	60.8	2.50	30	0.833	2.50	28.3	50.8
	L1-log	63.3	8.33	30	1.67	0.833	22.5	27.5
	L1-log-cos	64.2	8.33	30.8	0.833	0.833	21.7	27.5
(→) 1-NNA %	RMSE	55.4	96.3	93.4	100	100	84.6	60.0
	-PSNR	55.0	94.2	90.0	100	100	84.6	60.4
	-SSIM	57.5	96.3	96.7	100	100	86.3	61.7
	L1	58.8	100	95.4	100	100	92.5	80.0
	L1-log	98.3	100	92.1	100	100	100	100
	L1-log-cos	93.3	100	90.4	100	100	98.8	99.6
	FID (↓)	0.187	10.9	23.8	26.1	28.4	7.56	0.440

Table 4.5: Unconditional generative models performance on test set.



(a) Category averaged max and mean reflectance.



(b) K means on the maximum reflectance of red channel.

Figure 3.7: MERL BRDF category statistical analysis.

FreNBRDF: A Frequency-Rectified Neural Material RepresentationZheyuan Hu[†], [Chenliang Zhou](#)[†], [Cengiz Öztireli](#).*IEEE International Workshop on Machine Learning for Signal Processing (MLSP), 2025.*2024 - 2025 † *Computer Graphics (real-world materials), Frequency Rectification (Spherical Harmonics).*Evolved from my individual project in the [Machine Visual Perception](#) module (rank 2/15).

Supervised by my supervisor [Chenliang Zhou](#), I investigate the frequency information of materials, i.e. BRDF, and how to utilize it in the material fitting, reconstruction and editing. The first task is derived from a [set encoder](#) with permutation invariance and flexibility of input size. Then, we measure its capability in material linear interpolation editing.

We observe that the naive NBRDF autoencoder pipeline described above lacks the frequency information between the two NBRDFs, which might aid NBRDF weight distribution learning. I propose different methods to decompose BRDF three-dimensional inputs into those in unit sphere. Spherical harmonics is leveraged to express per-channel BRDF as a linear combination of orthonormal base functions, which captures the frequency information of BRDF.

This framework enhances fidelity, adaptability, and efficiency. Extensive experiments demonstrate that FreNBRDF improves the accuracy and robustness of material appearance reconstruction and editing compared to state-of-the-art baselines, enabling more structured and interpretable downstream tasks and applications.

Introduction and Motivation

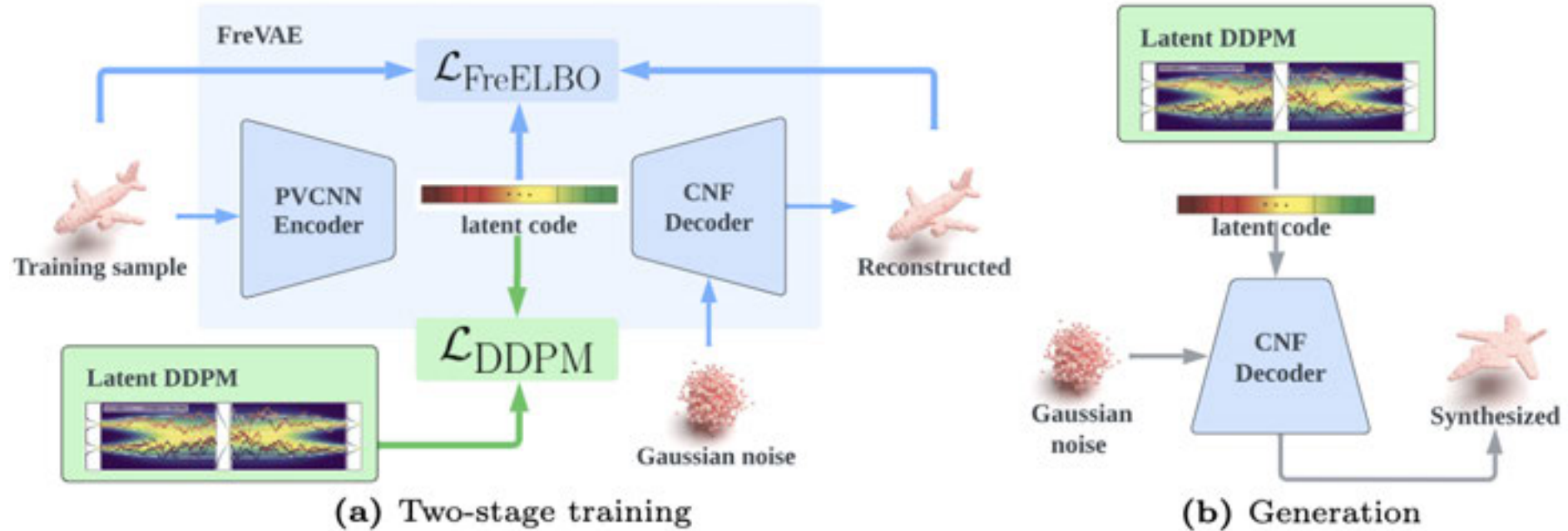
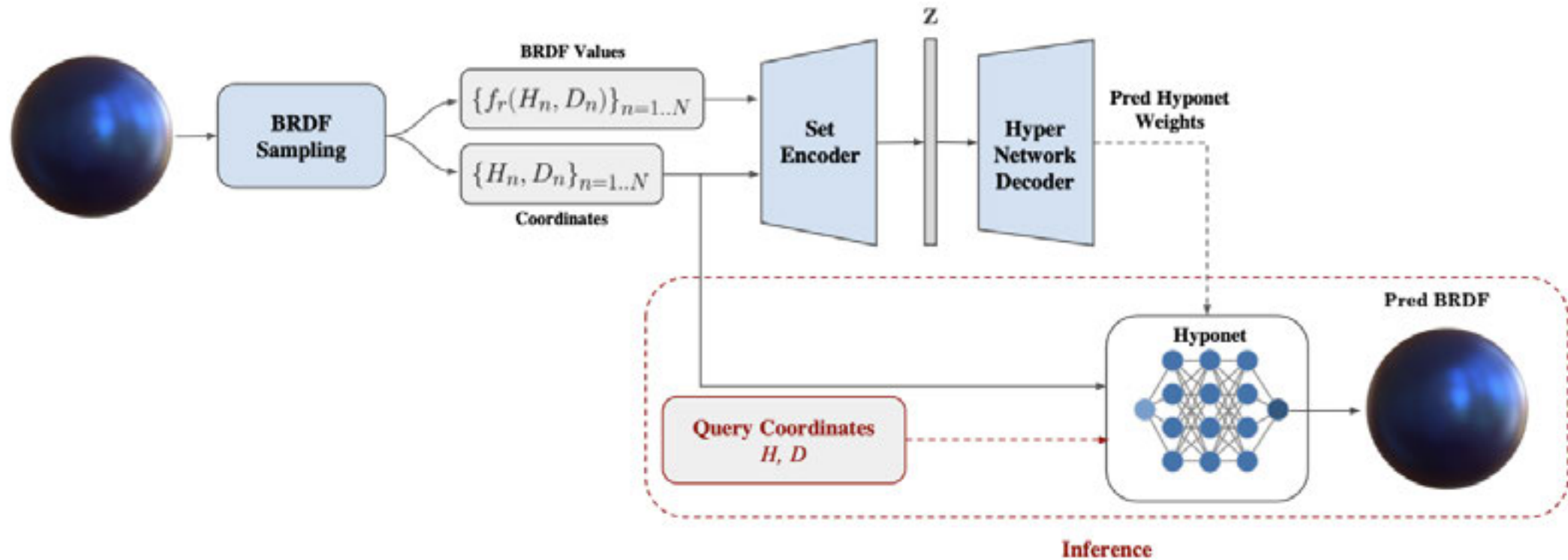


Fig. 2: FrePolad is architected as a point cloud VAE, with an embedded latent DDPM to represent the latent distribution. (a) Two-stage training: in the first stage (blue), the VAE is optimized to maximize the FreELBO Eq. (15) with a standard Gaussian prior; in the second stage (green), while fixing the VAE, the latent DDPM is trained to model the latent distribution. (b) Generation: conditioned on a shape latent sampled from the DDPM, the CNF decoder transforms a Gaussian noise input into a synthesized shape.

Baseline material reconstruction method



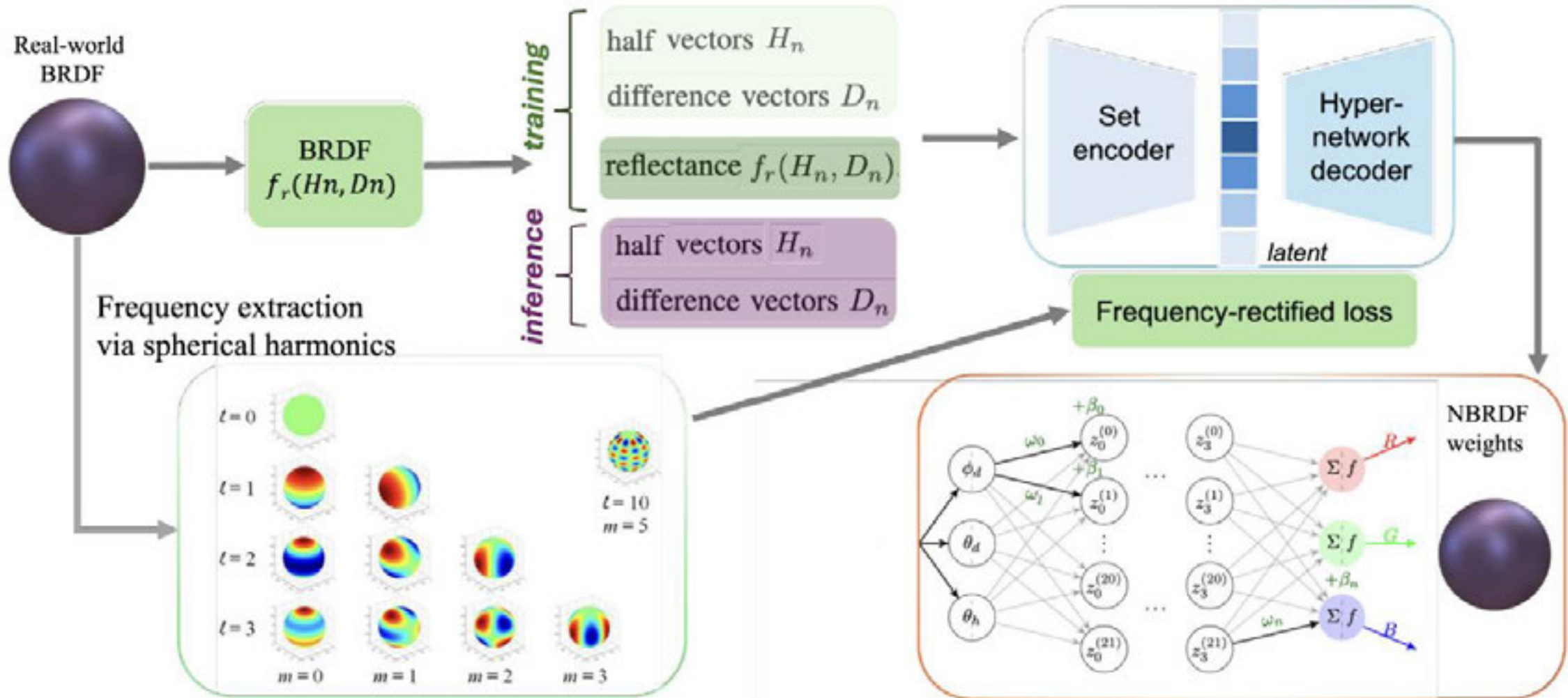


Fig. 1. Overview of FreNBRDF architecture.

*log-relative
mapping*

$$f_r \mapsto \ln \left(\frac{f_r + \epsilon}{f_{\text{ref}} + \epsilon} \right)$$

$$\begin{aligned} \mathcal{L}_{\text{rec}}(f_r, f'_r) := & \frac{1}{|P|} \sum_{i=1}^{|P|} |(f_r(H_i, D_i) - f'_r(H_i, D_i)) \cos \theta_i| \\ & + \lambda_1 \sum_{i=1}^W w_i^2 + \lambda_2 \sum_{j=1}^Z z_j^2, \end{aligned} \quad (1)$$

[HyperBRDF-ECCV'24](#)
(reconstruction loss)

Spherical harmonics

$$f_r(\theta, \varphi) = \sum_{l=0}^{\infty} \sum_{m=-l}^l c_{l,m} G_{l,m}(\theta, \varphi); \quad (8)$$

$$c_{l,m} = \int_0^{2\pi} \int_0^{\pi} f_r(\theta, \phi) \overline{G_{l,m}(\theta, \phi)} \sin \theta d\theta d\phi. \quad (9)$$

$$\mathcal{L}_{\text{fre}}(f_r, f'_r) := \frac{1}{|P|} \sum_{i=1}^{|P|} \|c_{l,m} - c'_{l,m}\|^2 \quad (10)$$

Ours
(frequency-rectified loss)

CHOrD: Generation of Collision-Free, House-Scale, and Organized Digital Twins for 3D Indoor Scenes with Controllable Floor Plans and Optimal Layouts

Chong Su[†], Yingbin Fu[†], Zheyuan Hu, Jing Yang, Param Hanji, Shaojun Wang, Xuan Zhao, Cengiz Öztireli, Fangcheng Zhong.

2025.2 | *Indoor Scene Synthesis, Generative Models, Digital Twin Generation.*



A. Spatially coherent 3D digital twin generation

Mentored by Dr Fangcheng Zhong, we investigated how 2D floor plans generation can be integrated with 3D indoor scene synthesis, for lower collision rate and higher fidelity. In addition, a new real-world indoor scenes dataset will be released.

I am responsible for reproducing the prior art DiffuScene, CVPR24 and InstructScene, ICLR24 Spotlight. Meanwhile, I investigated the YOLO-v8 Oriented Bounding Box (OBB) detection error and proposed alternative methods, e.g. Rotated Bounding Box (RBB).

The former DiffuScene is based on applying diffusion model for parametrized objects (location, size, orientation, class, shape code). For greater instruction control, the latter InstructScene follows a two-stage procedure. A semantic graph, with class label, (additional) pairwise spatial relationship and quantized features, is constructed from the prior semantic input. After that, a separate 3D decoder determines the exact layout (location, size, orientation). For details, please refer to retraining InstructScene on our CF-GISS dataset.

Instruct Scene on CF-GISS dataset

Peter HU, 18 Feb 2025

Project page: <https://chenguolin.github.io/projects/InstructScene/>

Github: <https://github.com/chenguolin/InstructScene>

arXiv: <https://arxiv.org/abs/2402.04717>

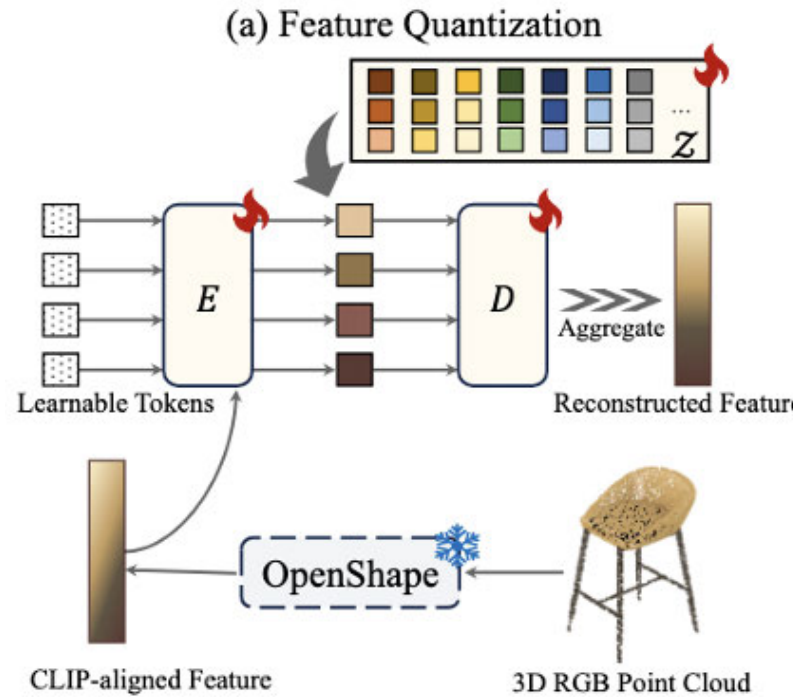


Table 5: Prompt for ChatGPT to refine raw object descriptions.

Given a description of furniture from a captioning model and its ground-truth category, please combine their information and generate a new short description in one line. The provided category must be the descriptive subject of the new description. The new description should be as short and concise as possible, encoded in ASCII. Do not describe the background and counting numbers. Do not describe size like 'small', 'large', etc. Do not include descriptions like 'a 3D model', 'a 3D image', 'a 3D printed', etc. Descriptions such as color, shape and material are very important, you should include them. If the old description is already good enough, you can just copy it. If the old description is meaningless, you can just only include the category. For example: Given 'a 3D image of a brown sofa with four wooden legs' and 'multi-seat sofa', you should return: a brown multi-seat sofa with wooden legs. Given 'a pendant lamp with six hanging balls on the white background' and 'pendant lamp', you should return: a pendant lamp with hanging balls. Given 'a black and brown chair with a floral pattern' and 'armchair', you should return: a black and brown floral armchair. The above examples indicate that you should delete the redundant words in the old description, such as '3D image', 'four', 'six' and 'white background', and you must include the category name as the subject in the new description. The old descriptions is '{BLIP caption}', its category is '{ground-truth category}', the new descriptions should be:

Instruction prompts for object features fVQ-VAE

(0) Scene description text y
=> semantic-features

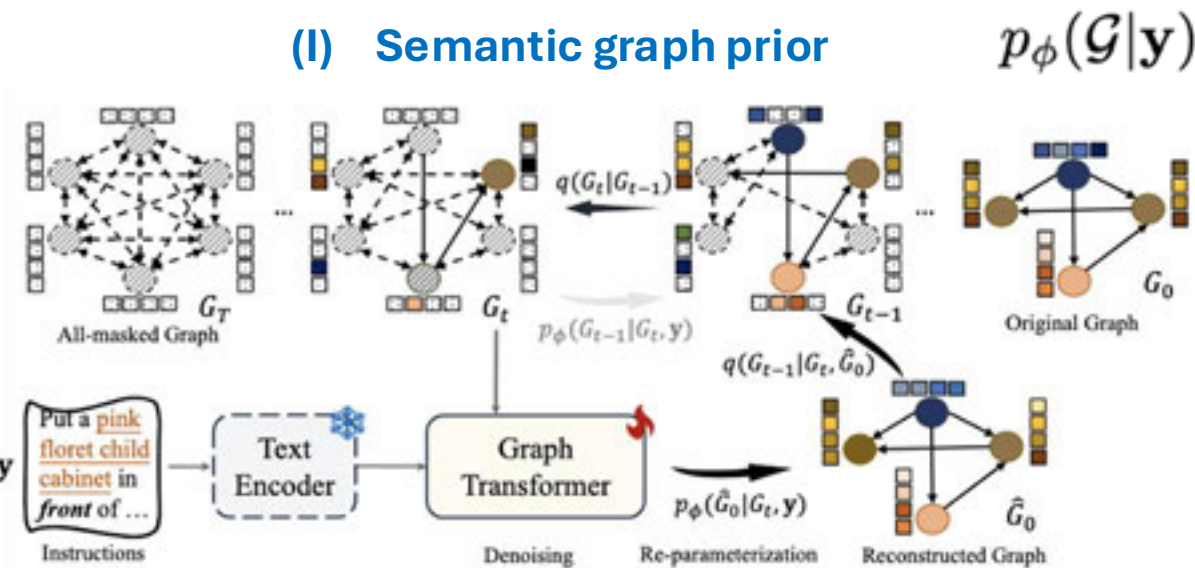
$$p_{\phi, \theta}(\mathcal{S}|\mathbf{y}) = p_{\phi, \theta}(\mathcal{S}, \mathcal{G}|\mathbf{y}) = \underbrace{p_{\phi}(\mathcal{G}|\mathbf{y})}_{\text{Semantic Graph Prior}} p_{\theta}(\mathcal{S}|\mathcal{G}). \quad (2)$$

(0) Scene text \mathbf{y}
 \Rightarrow semantic-features

$\rightarrow \mathbf{I} \dots$ Graph: {
 semantic-features: None,
 objs_id: [2,3,5,13,13],
 objs_mask: [1,1,1,0,0], edges: [N^2],
 (->) boxes: [position^3, scale^3, cos\theta, sin \theta]
 }

Final Scene

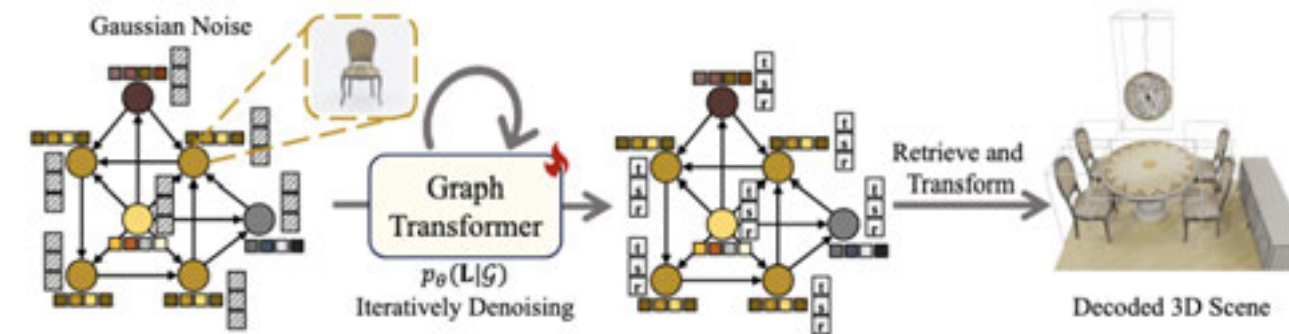
(I) Semantic graph prior



$(\mathbf{G}_0, \mathbf{y}) \dots (\mathbf{G}_T, \mathbf{y}) \dots (\mathbf{G}_{0^*}, \mathbf{y})$

\mathbf{G} : Graph without boxes

$p_{\theta}(\mathcal{S}|\mathcal{G})$ (II) 3D layout decoder



$\mathbf{G}_{0^*} \Rightarrow$ boxes

edge_{i,j}: geometric relationship

Relationship	Rule
Left of	$(\theta_{so} \geq \frac{3\pi}{4} \text{ or } \theta_{so} < -\frac{3\pi}{4}) \text{ and } 1 < d(s, o) \leq 3$
Right of	$-\frac{\pi}{4} \leq \theta_{so} < \frac{\pi}{4} \text{ and } 1 < d(s, o) \leq 3$
In front of	$\frac{\pi}{4} \leq \theta_{so} < \frac{3\pi}{4} \text{ and } 1 < d(s, o) \leq 3$
Behind	$-\frac{3\pi}{4} \leq \theta_{so} < -\frac{\pi}{4} \text{ and } 1 < d(s, o) \leq 3$
Closely left of	$(\theta_{so} \geq \frac{3\pi}{4} \text{ or } \theta_{so} < -\frac{3\pi}{4}) \text{ and } d(s, o) \leq 1$
Closely right of	$-\frac{\pi}{4} \leq \theta_{so} < \frac{\pi}{4} \text{ and } d(s, o) \leq 1$
Closely in front of	$\frac{\pi}{4} \leq \theta_{so} < \frac{3\pi}{4} \text{ and } d(s, o) \leq 1$
Closely behind	$-\frac{3\pi}{4} \leq \theta_{so} < -\frac{\pi}{4} \text{ and } d(s, o) \leq 1$
Above	$(\text{Center}_{Z_s} - \text{Center}_{Z_o}) > (\text{Height}_s + \text{Height}_o)/2$ and $(\text{Inside}(s, o) \text{ or } \text{Inside}(o, s))$
Below	$(\text{Center}_{Z_o} - \text{Center}_{Z_s}) > (\text{Height}_s + \text{Height}_o)/2$ and $(\text{Inside}(s, o) \text{ or } \text{Inside}(o, s))$
None	$d(s, o) > 3$

We set the threshold from 1~3 to **150~450** after observation, leaving the rest same as theirs.

Bedroom inference results



Livingroom inference results





Thank you~

Peter HU, 18 Feb 2025

Project page: <https://chenguolin.github.io/projects/InstructScene/>

Github: <https://github.com/chenguolin/InstructScene>

arXiv: <https://arxiv.org/abs/2402.04717>

Industry research

@ **Industry Research Center, Cambridge Science Park**, UK. Details: 

Research Engineer: **Graphics Algorithm/GPU Architecture**

Jun 2023-Jan 2024

- Linear Algebra, Convolution (Bilateral Filter Kernel on Monte Carlo Samples using GBuffer), spatial-temporal locality.
- NN (PyTorch): Train (lr decay, shuffle data 5GB+, dropout) and Infer (conservative loss), 3D Data Encoding, etc.
- Graphics: Key developer for **Ray Tracing simulation** (OpenGL, GLSL, OpenMP, CMake). Host **sharing sessions**.
- Performance Engineer / Data structure design, targeting micro-benchmarks (performance counters, cache hit rate, etc.)
- Supervised by PhD graduate, senior AI researcher and senior GPU Architects.

Research Intern: **CPU Architecture**

Jun-Oct 2023

- **Review** of CPU Scheduling, DVFS policy, Idle Management in terms of energy efficiency. Convex Optimisation, Duality, LP, Pareto Optimality, Stanford CVX, Online Algorithms, Competitive Analysis, Disjoint Set Union-find, etc.
- Set up simulation, event-driven architecture with state machine, taking in runtime profiled task model. Compare different algorithms w.r.t complexity, performance, energy (temperature, thermal), Memory Contention, floor-plan, applications. Python (Numpy, Matplotlib, Networkx, Pandas, DAG, TopologicalSorter, etc).

Software Engineer: **GPU Driver**

Dec 2022-May 2023

- GPU industry workflow, Linux, Vulkan; GPU driver and verification, Game Engines (UE4), shader debug (RenderDoc).
- Introducing independent full [automation tools](#) in the CI/CD, reducing error rate to nearly 0.

For details, please refer to <https://peterhuistyping.github.io/#!/research/#industryResearch>

Industry research

Hash-Based Ray Path Prediction:

Skipping BVH Traversal Computation by Exploiting Ray Locality, EUROGRAPHICS 2019.

Neural Intersection Function, High-Performance Graphics 2023.

Ray Classification for Accelerated BVH Traversal, Eurographics Symposium on Rendering 2019.

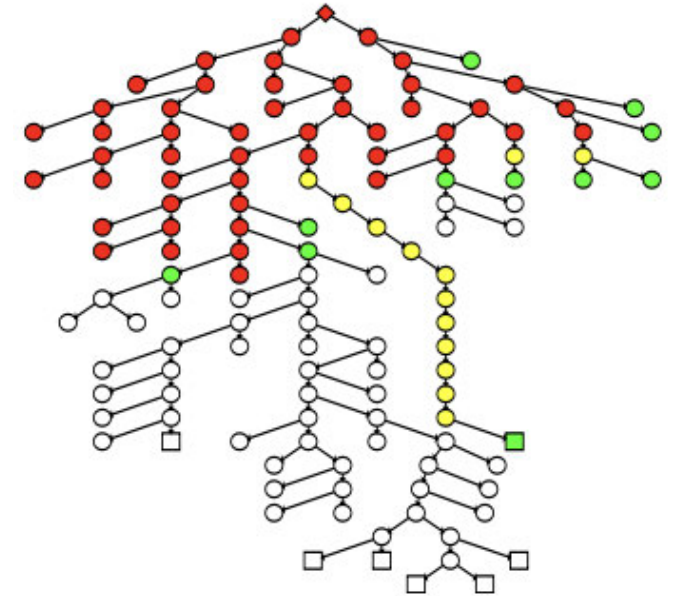
The red nodes are visited only by the standard traversal from the root.

The green nodes are entry points of our traversal algorithm.

The yellow nodes are visited by neither traversal method and only denote the path to candidate list elements.

The white nodes are visited by both types of traversal.

In this case, our method skips 44% of nodes, both on the path from the BVH root and in lateral branches.



For details, please refer to <https://peterhuisting.github.io/#/research/#industryResearch>

Star of Cambridge



Awarded to

Mihai Florin Barbu 00540052 & Zheyuan Hu 50031312

Mihai Florin Barbu and Zheyuan Hu proposed the ray-prediction algorithm. According to the test results, the ray intersection latency in reflection scenarios can be reduced by 33%, RTU energy consumption can be reduced by 15%, or RTU throughput can be improved by 20%. The results achieved are recognized by the hardware team. This algorithm will be the official delivery technology of the [redacted] project. They have demonstrated strong algorithmic capabilities and have shown typical examples of cross-team collaboration.

Well done and congratulations!

周书亚

zhou shuyue

Team Leader

Cambridge Research Center

05/12/2023

Date

Selected projects

Gold Medal, 3D data compression algorithm @ UK Tech Arena [*C and C++, Compression, Concurrent*]

2022.10.20-2022.11.26 | *Issued by Huawei UK R&D.*

Won £7000 prize and organized a team of 4.

Learning and researching from scratch in a month, digesting lots of papers and source code available, like RFC1951, LZSS.

Responsible for LZSS algorithm and improvements. Brainstorming with team members. Demonstrate my presentation skills.



Top 2 Team, Maritime Data Science @ Mercuria Hackathon [*Python, Data Analysis, Route-Planning*]

2022.12.16-2022.12.18 | *Issued by Mercuria Energy Trading, Switzerland.*

Using data analysis to accelerate the energy transition and reduce the carbon emissions of the maritime industry.

Python Data Analysis, Route Plotting.

Great Team Work, Collaboration.

Networking with senior engineers, excellent undergraduate, Master and PhD students from all around the Europe.



Selected projects

Literal Run

000 L_4-L_0 | ...

length = 9

I am Sam\n

Sam

I am

\nThat

08 I am Sam10(\n)

20 03 00 20()

40 0C

04 10(\n) That

Note:

00 20 : 0000 0000 (20)₁₀

000	L_4-L_0	...
000	00000	(20) ₁₀

Short Match

M_2-M_0 $W_{12}-W_8$ | W_7-W_0

I am Sam\n

Sam

I am

\nThat

08 I am Sam10(\n)

20 03 00 20()

40 0C

04 10(\n) That

Note:

20 03 : 0010 0000 0000 0011

M_2-M_0	$W_{12}-W_8$	W_7-W_0
001	00000	00000011

Relationship between Original File and Encoded File		
Original Match	→	Encoded Len
1-2 Bytes	→	1 Bytes (Literal Run)
3-8 Bytes	→	2 Bytes (Short Match)
9-264 Bytes	→	3 Bytes (Long Match)

By using the First 3 bits of the Literal Run at the beginning of the Input File as Flag:

```
level 1 000
level 2 100 *(ubyte*)OUTPUT_ |= (1 << 7);
level 3 010 *(ubyte*)OUTPUT_ |= (1 << 6);
level 4 001 *(ubyte*)OUTPUT_ |= (1 << 5);
```

Selected projects

Part 1 | Infinite Match Length

Level 2: $2^{13} = 8192$, i.e., $[0..(8191 - 1)]$

(11111 0xff) is used as flag for whether it's Extended Window.

Match type	Literal run	Short match	Long match
Decode[0]	000L ₄ -L ₀	M ₂ M ₁ M ₀ W ₁₂ -W ₈	111W ₁₂ -W ₈
Decode[1]		W ₇ -W ₀	M _{n+7} -M _n
Decode[..]			M ₇ -M ₀
Decode[2]			W ₇ -W ₀

Extra Windows Size

Level 3: $2^{16} = 65536$, i.e., $[8191 + (65535 - 1)..8191 + (65535 - 1) + (65535 - 1)]$.

(11111 0xff) is used as flag for End.

Match type	Literal run	Short match*	Long match*
Decode[0]	000L ₄ -L ₀	M ₂ M ₁ M ₀ 11111	111 11111
Decode[1]		11111111	M _{n+7} -M _n ...
Decode[2]		0xff	11111111
Decode[3]		0xff	0xff
Decode[4]		W ₁₅ -W ₈	0xff
Decode[5]		W ₇ -W ₀	W ₁₅ -W ₈
Decode[6]			W ₇ -W ₀

Part 2 | *Extended Window Size

Level 2: $2^{16} = 65536$, i.e., $[8191..8191 + (65535 - 1)]$.

(11111 0xff) is used as flag for End.

Match type	Literal run	Short match*	Long match*
Decode[0]	000L ₄ -L ₀	M ₂ M ₁ M ₀ 11111	111 11111
Decode[1]		11111111	M _{n+7} -M _n ...
Decode[..]			M ₇ -M ₀
Decode[2]		W ₁₅ -W ₈	11111111
Decode[3]		W ₇ -W ₀	W ₁₅ -W ₈
Decode[4]			W ₇ -W ₀

Ultra Windows Size

Level 4: $2^{16} = 65536$, i.e., $[8191 + (n - 1) * (65535 - 1)..8191 + n * (65535 - 1)]$.

(11111 0xff) is used as flag for End.

Match type	Literal run	Short match*	Long match*
Decode[0]	000L ₄ -L ₀	M ₂ M ₁ M ₀ 11111	111 11111
Decode[1]		11111111	M _{n+7} -M _n ...
Decode[2]		0xff	11111111
Decode[3]		0xff...	0xff
Decode[4]		W ₁₅ -W ₈	0xff...
Decode[5]		W ₇ -W ₀	W ₁₅ -W ₈
Decode[6]			W ₇ -W ₀

Selected projects

Direct Match

The data structure of Match Length is not efficient enough

Match type	Literal run	Short match	Long match
Decode[0]	000L ₄ -L ₀	M ₂ M ₁ M ₀ W ₁₂ -W ₈	111W ₁₂ -W ₈
Decode[1]		W ₇ -W ₀	...M ₇ -M ₀
Decode[2]			W ₇ -W ₀

Direct Match

Level 5: A better solution to Match Length Data Structure

Reserve Flag 110 in Short Match for the new Direct match; Same for the Extended Window Part

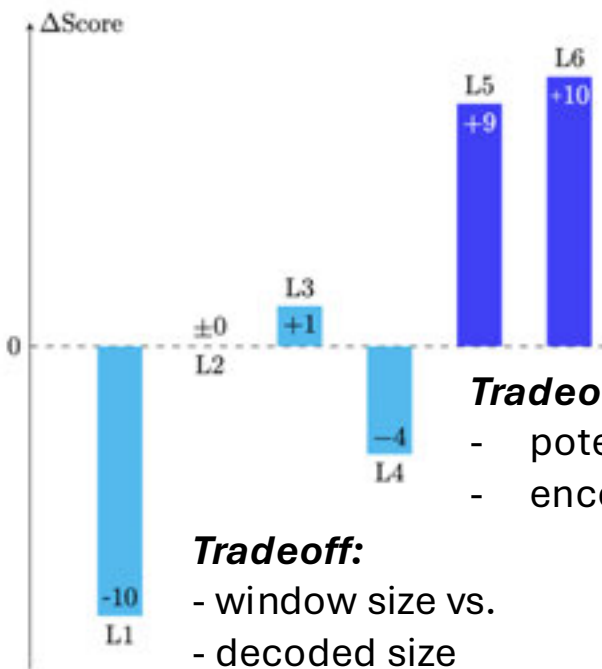
Match type	Long match	Direct match	Long match*
Decode[0]	111 W ₁₂ -W ₈	110 W ₁₂ -W ₈	111 W ₁₂ -W ₈
Decode[1]	M ₇ -M ₀	D ₁₅ -D ₈	11111111
Decode[2]	W ₇ -W ₀	D ₇ -D ₀	...M ₇ -M ₀
Decode[3]		W ₇ -W ₀	W ₇ -W ₀

Direct Long Match

Level 6: Introduce a 4 Bytes threshold D₃₁ - D₀ for Match Length

Reserve Flag 0x00 in the first M_{n+7} - M₀ for the Max of Direct Long Match; Same for the Extended Window Part

Match type	Long match	Direct Long match*
Decode[0]	111 W ₁₂ -W ₈	111 W ₁₂ -W ₈
Decode[1]	11111111	11111111
Decode[2]	...M ₇ -M ₀	00000000
Decode[3]	W ₇ -W ₀	D ₃₁ - D ₀
Decode[4]		W ₇ -W ₀



Tradeoff:

- potential Match length
- encoded length

Tradeoff:

- window size vs.
- decoded size

Selected projects

Lots of Low Level Operations:

- i. Using **pointers** of C without STL from C++.
- ii. **Bitwise Operators**, dealing with Binary Numbers.
- iii. **Hash Table**, storing sequences that appeared before.
- iiii. **Fixed width integer types**, like uint8_t, uint32_t
To achieve Faster Decompression Speed

Concurrent Decompression: Parallelization

- i. Divide the Input into n subgroups(via 2D arrays.)
- ii. Use Multithreading to decode the n subgroups and put into the same vector.

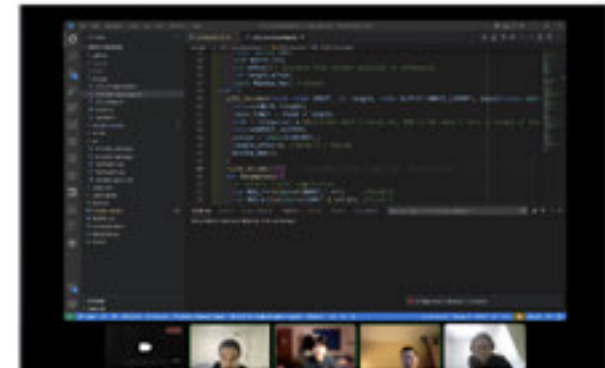
```
vector<thread> threads;
auto lambda=[&](int tid){
    Decompress(buffer3+tid*each_numbytes,
        compressed_size[tid],
        buffer4+tid*each_numbytes);
};
for(int tid=0;tid<num_threads;tid++){
    threads.push_back(thread(lambda,tid));
}
for(int tid=0;tid<num_threads;tid++){
    threads[tid].join();
}
```

Compiler Optimizations: Branch Prediction

```
if (len > 264 - 2)[[unlikely]]
//Max of Match Length 256+8
while (len > 264 - 2) {
    output[index++] = (7 << 5) + (distance >> 8);
    output[index++] = 264 - 2 - 7 - 2;
    output[index++] = (distance & 0xff);
    len -= 264 - 2;
}
cmp =(__builtin_expect(!(distance < windows_size), 1))
? read_uint32(ref) & 0xffffffff : 0x1000000;
return(sequence==cmp);
```

Ending: Reflection and Acknowledgement

Core values: Collaboration, Innovation, Continuous Learning
Thank you! Mentors, Huawei, and other Participants



Selected projects

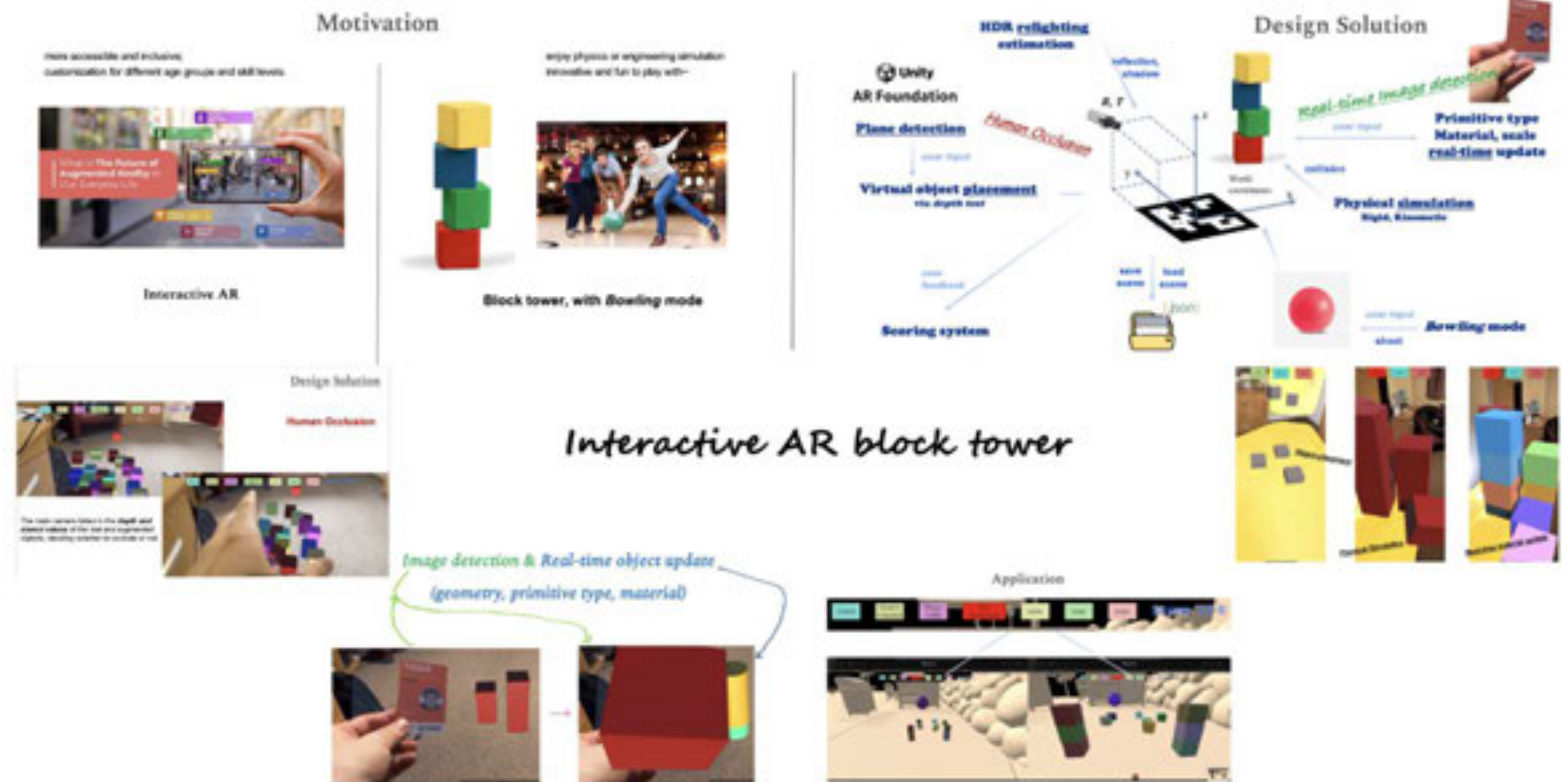
Interactive AR block tower [Unity, AR foundation, C#]

2025.1.20-2025.3.20

Extended Reality (XR) module project

Unity video based AR app

Conclusion



Thank you very much!

Looking forward to

- potential PhD position,
- future collaboration,
- and keeping connected.

Homepage: <https://peterhuistyping.github.io/>

Github: <https://github.com/PeterHUistyping>

CV: https://peterhuistyping.github.io/asset/doc/CV_PeterHU.pdf



Peter/Zheyuan HU