CHULA ENGINEERING
Innovation toward Sustainability | ACTNOW

SkyGPT revisited

Advisor Dr. Suwichaya Suwanwimolkul

Mr. Tiranut Kanjanatunyarut 6530141421

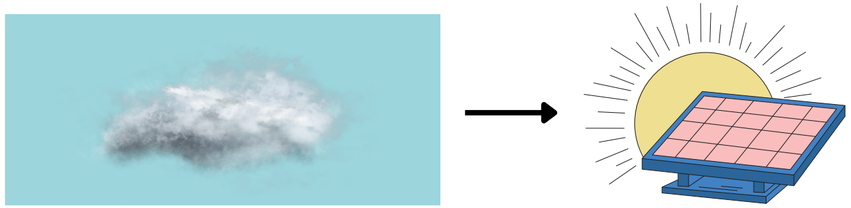
Mr. Wachirawit Piyaprapapan 6530349721

Introduction

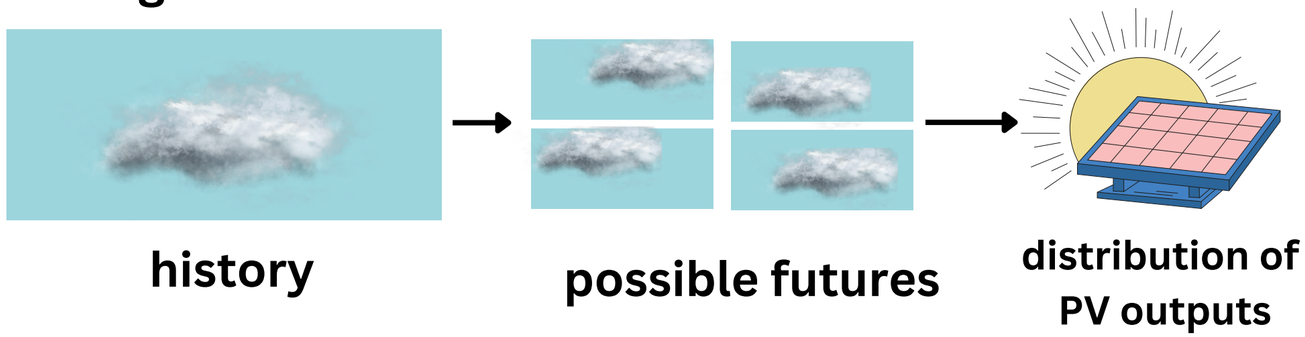
Renewable energy sources, such as solar photovoltaics (PV), will be the key component for future power systems. One of the challenges for deployment of PV, especially at a large scale, is unstable power generation.

objective: To forecast photovoltaic power output using historical cloud image sequences.

Traditional:



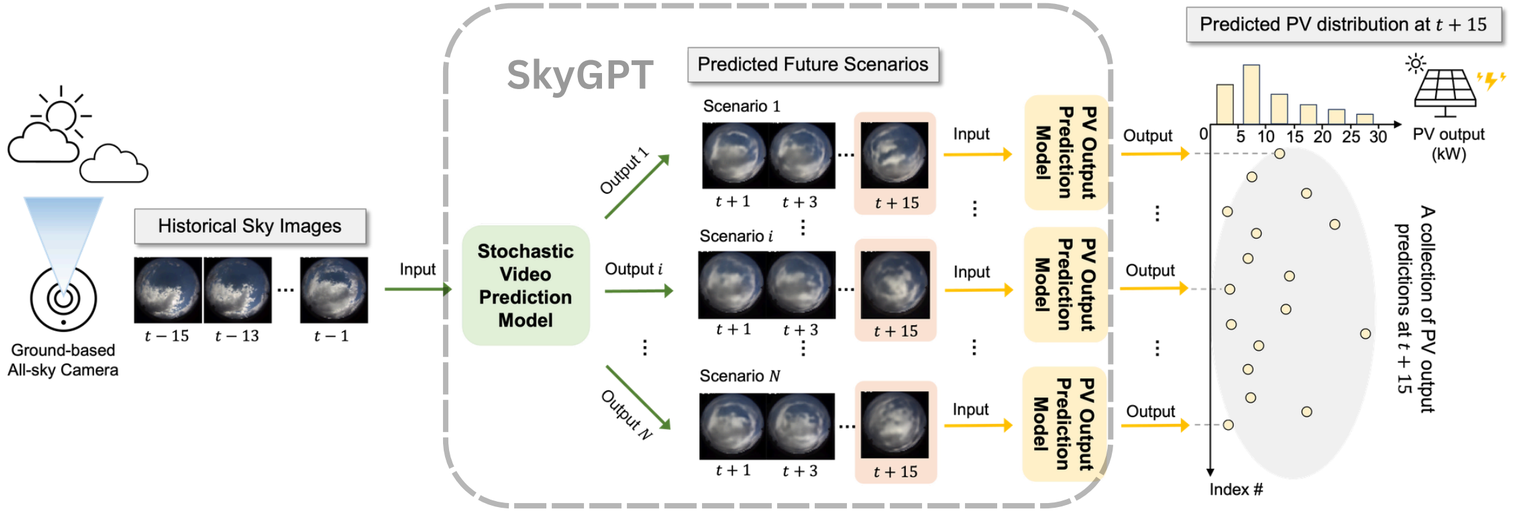
With genarative models :



Why Generative models? Predicting cloud motion requires capturing complex uncertainty. Generative models allow us to effectively sample the probabilistic distribution of cloud movements, leading to better forecasting performance.

What is SkyGPT?

A hybrid framework combining **generative AI(VideoGPT)** and **physics-informed dynamics (PhyDNet)** for 15-minute-ahead **stochastic sky video prediction**, followed by **probabilistic photovoltaic power forecasting** using the predicted sky images.



10-25% improvement in prediction reliability and sharpness is observed with SkyGPT when compared to baselines and other benchmark models (VideoGPT, PhyDNet)**, as claimed** in the research paper.**

SkyGPT overview

In our research scope, we will focus only on **the stochastic sky video prediction** part

The stochastic sky video prediction components

1. VQ-VAE

Compress high-dimensional sky video sequences into discrete codebook representations

2. Phy-transformer

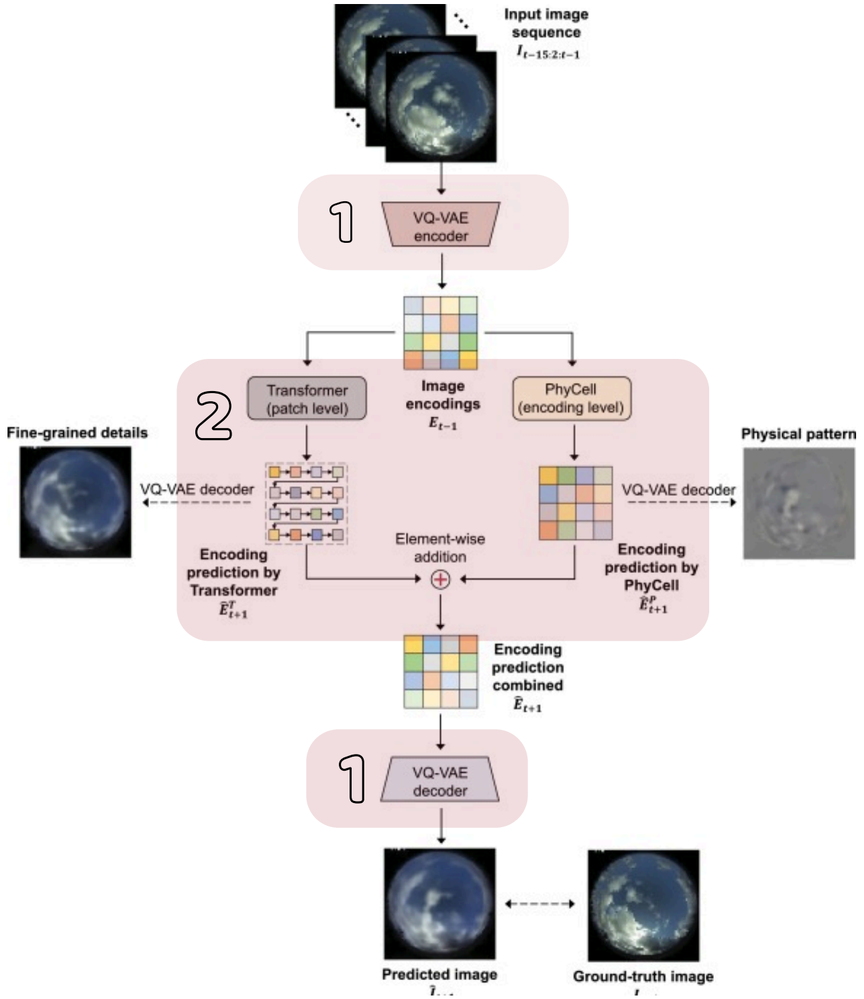
2.1. PhyCell (PhyDNet modified)

Capture global motion patterns following atmospheric physics laws

2.2. VideoGPT transformer

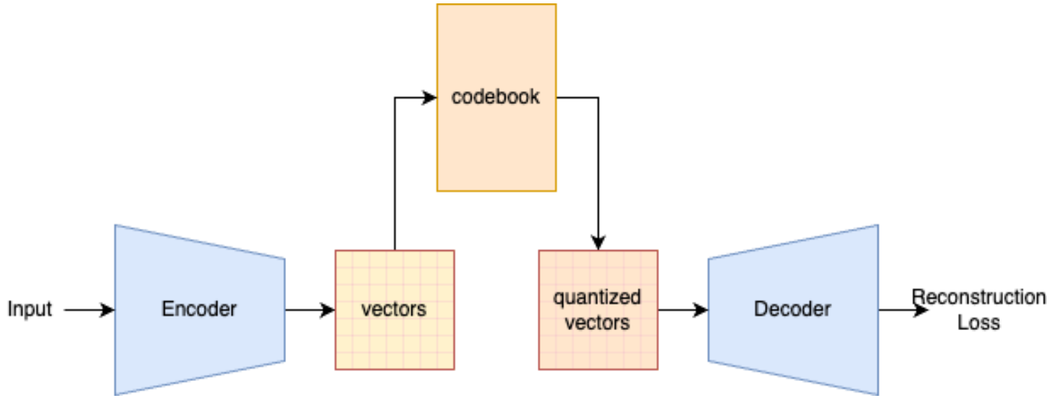
Stochastic sky video prediction from previous sky images, gives patch level encoded prediciton

****Each components are trained separately****



VQ-VAE

Video prediction favours discrete latent vectors cause it make the video less blurry (instead, sharp and fast)



- **Encoder** quantized each area by looking at each position in image, find **nearest** codebook vectors and replaced it
- **Quantized vectors** are passed to both the PhyCell module and Transformer module for temporal modeling and future prediction
- **Decoder** learn to reconstruct the whole picture by reading the embeddings and plugging the mapped codebook vectors

This make the image constructed by discrete codebook vector not soft mix of them

VQ-VAE objective function

- **reconstruction loss** → how accurate the reconstrcted output differ from real input
- **codebook loss** → how shift from output embeddings to codebook embeddings
- **commitment loss** → how fluctuation when choosing codebook embeddeds

$$\mathcal{L} = \beta_{recon} \underbrace{\|x - \hat{x}\|_2^2}_{\text{reconstruction}} + \beta_{code} \underbrace{\|\text{sg}[z_e(x)] - z_q\|_2^2}_{\text{codebook loss}} + \beta_{com} \underbrace{\|z_e(x) - \text{sg}[z_q]\|_2^2}_{\text{commitment loss}}$$

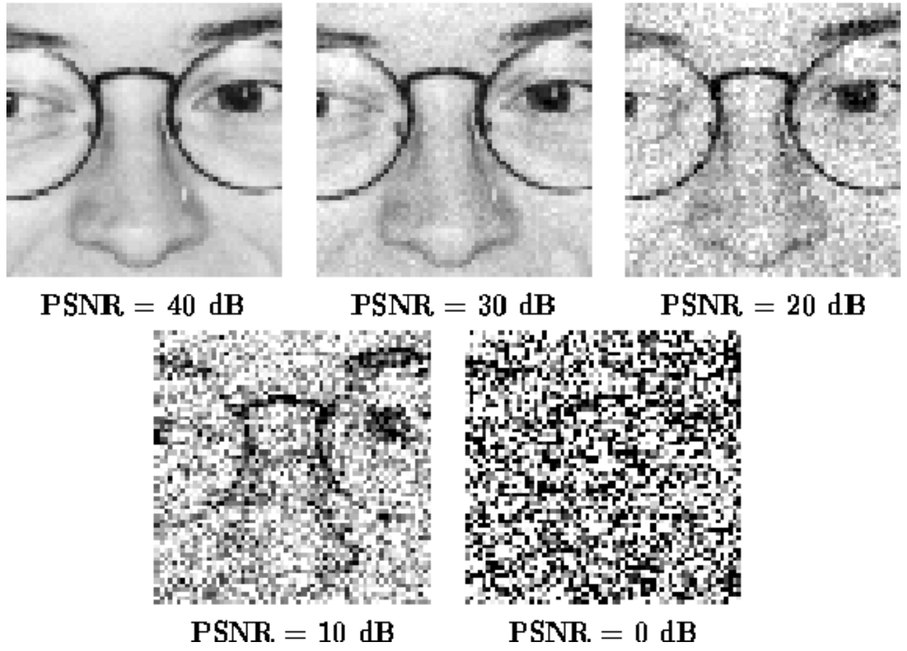
x : input
$\hat{x}$: reconstruction
$z_e(x)$: encoder output
z_q : nearest codebook vector
$\text{sg}[\cdot]$: stop-gradient operator
β : commitment weight

With stop gradient :

- Each piece has clear gradient flow rules.
  - **Encoder** cares commitment and reconstruction loss.
  - **Codebook** cares codebook and reconstruction loss.
  - **Decoder** cares reconstruction loss only.

VQ-VAE evlauation metric : PSNR

PSNR: measures the visual quality of reconstructions by comparing them to the original images.



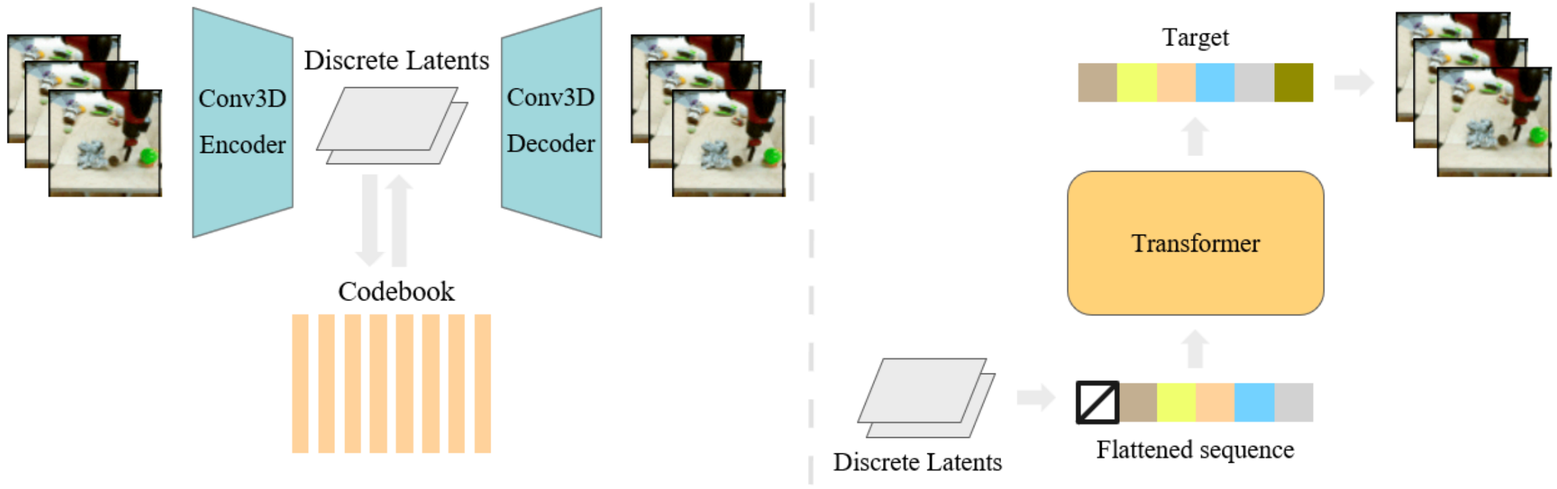
Peak Signal-to-Noise Ratio (PSNR)

$$\text{PSNR} = 10 \log_{10} \left(\frac{\text{MAX}^2}{\text{MSE}} \right), \qquad \text{MAX} = 255 \qquad \text{MSE} = \frac{1}{N} \sum_{i=1}^{N} (I_i - \hat{I}_i)^2,$$

$$\begin{aligned} & N = \text{total number of pixels}, \\ \textit{where} \quad & I_i = \text{original pixel value at position } i, \\ & \hat{I}_i = \text{reconstructed pixel value at position } i. \end{aligned}$$

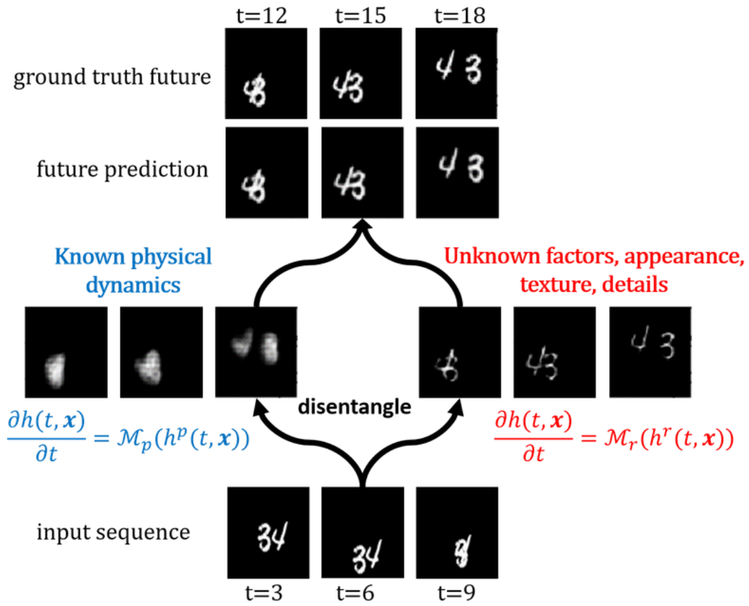
VideoGPT transformer

- cover **fine-grained details** which is unknown to PhyCell.
- Uses **VQ-VAE** to discretize video frames into compressed latent codes.
- Autoregressively predicts the next tokens one by one using a GPT-like architecture.
- get a fine-grained prediction which we used to combined with the physic-based one.
- It will make **a crisp image sequence** with realistic physic movements clouds altogether.



PhyDNet

- Split the latent state into **two branches:** a physical branch and a residual (non-physical) branch.
- **Physical Branch (PhyCell)**: Introduces a physically-constrained recurrent cell to model Partial Differential Equations (PDEs), ensuring the model sticks to physical laws and improving generalization.
- **Residual Branch (ConvLSTM):** Employs a generic RNN (specifically ConvLSTM) to handle unknown phenomena and residual details that do not correspond to prior physical models.

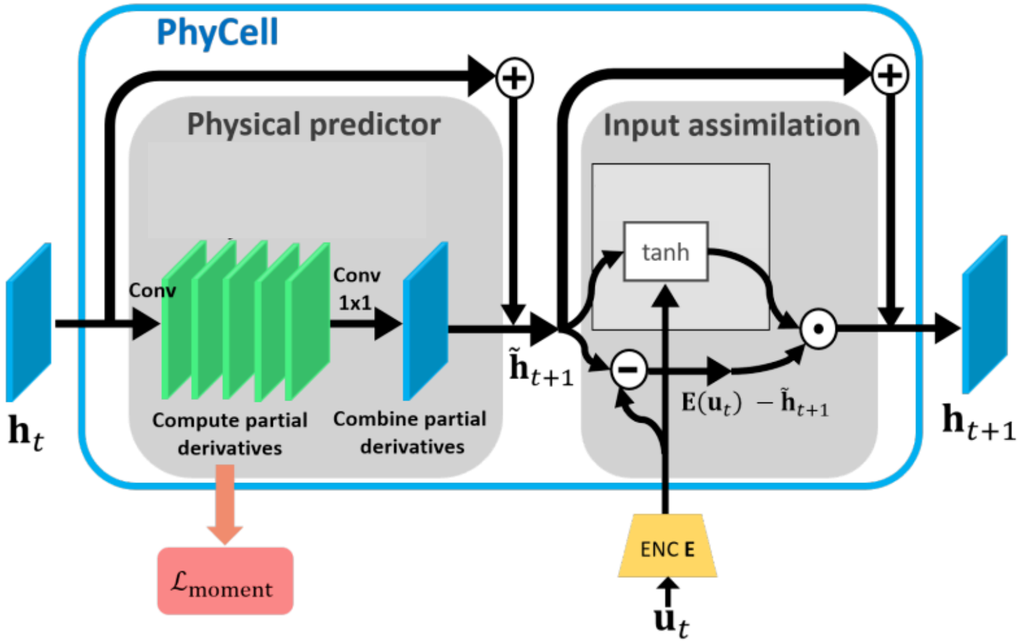


$$\frac{\partial h(t,x)}{\partial t} = M_p(h^p, u) + M_r(h^r, u)$$

M_p : physical dynamic in latent space
M_r : residual dynamic in latent space
h_p : physical components
h_r : residual components
u : frame of video at a time and spartial coordinate

PhyCell (physical branch of PhyDNet)

- Enforces physics-based constraints through **linear partial differential equations (PDE)** at the image embedding level
- Uses convolution operations to approximate spatial derivatives representing atmospheric physics
- demonstrates effectiveness in capturing the general trend of cloud motion, **but the generated images are quite blurry**
- the method is **similar** to Euler forward in digital control system



PhyCell recurrent cell implements a two-steps scheme

1. **Physical predictor**

   physical prediction with convolutions for approximating and combining spatial derivatives from previous latent state

2. **Input assimilation**

   correction of latent physical dynamics driven by observed data (input)

Method and Result

Modified loss funciton

update encoder/decoder — update codebook — update encoder

$$L_{original} = \beta_{recon} \underbrace{\|x - \hat{x}\|_2^2}_{\text{reconstruction}} + \beta_{code} \underbrace{\|\text{sg}[z_e(x)] - z_q\|_2^2}_{\text{codebook loss}} + \beta_{com} \underbrace{\|z_e(x) - \text{sg}[z_q]\|_2^2}_{\text{commitment loss}}$$

$$L_{\text{modified}} = \beta_{\text{Charbonnier}} L_{\text{Charbonnier}} + \beta_{\text{commitment}} L_{\text{commitment}} + \beta_{\text{perceptual}} L_{\text{perceptual}}$$

- Replace MSE with **Charbonnier loss**
- Replace codebook vectors Z_q with **EMA Update** decoupled with loss function
- Add **Perceptual loss**

Modified Loss Funciton : EMA Update van den Oord et al. (2017)

$$n_i^{(t)} = \sum_{j=1}^{B} \mathbf{1}[q(z_e(x_j)) = i]$$

$$s_i^{(t)} = \sum_{j=1}^{B} \mathbf{1}[q(z_e(x_j)) = i] z_e(x_j)$$

$$N_i^{(t)} \leftarrow \gamma N_i^{(t-1)} + (1-\gamma) n_i^{(t)}$$

$$m_i^{(t)} \leftarrow \gamma m_i^{(t-1)} + (1-\gamma) s_i^{(t)}$$

$$\tilde{N}_i^{(t)} = \frac{N_i^{(t)} + \epsilon}{\sum_{k=1}^{K} N_k^{(t)} + K\epsilon} \sum_{k=1}^{K} N_k^{(t)}$$

$$e_i^{(t)} \leftarrow \frac{m_i^{(t)}}{\tilde{N}_i^{(t)}}$$

i : codebook index
K : number of codebook entries
B : batch size
$z_e(x_j)$: encoder output for sample x_j
$q(z_e(x_j))$: nearest codebook index assigned to $z_e(x_j)$
$n_i^{(t)}$: number of vectors assigned to code i in batch t
$s_i^{(t)}$: sum of encoder vectors assigned to code i in batch t
$N_i^{(t)}$: EMA usage count for code i
$m_i^{(t)}$: EMA sum of encoder vectors assigned to code i
$\tilde{N}_i^{(t)}$: smoothed normalized EMA usage count
$e_i^{(t)}$: updated codebook embedding for code i

For each codebook entry, the model keeps track of two moving statistics: how often that entry is selected, and the average encoder output assigned to it. Instead of updating the codebook directly with noisy gradient steps, the embedding is updated **using a smoothed average of the encoder vectors that choose it**. In simple terms, each codebook entry gradually moves toward the group of latent vectors it represents.

Modified Loss Funciton : EMA Update van den Oord et al. (2017)

$$Hyperparameters \quad \begin{matrix} \gamma : \text{EMA decay} \\ \epsilon : \text{smoothing constant} \\ \text{Restart threshold} \end{matrix}$$

The smoothing term prevents unstable updates when a codebook entry is rarely selected or not selected at all. This helps avoid division-by-zero issues and keeps rare entries numerically stable. In addition, some implementations reset dead codebook entries by replacing them with random encoder vectors when they are used too rarely. This reset strategy can help reduce codebook collapse, but it is only a practical heuristic and does not guarantee that all codebook entries will be used equally.

Modified Loss Funciton : Perceptual Loss

Johnson et al. (2016).

$$\mathcal{L}_{\text{perceptual}} = \sum_{l} \lambda_l \left\| \phi_l(x) - \phi_l(\hat{x}) \right\|_2^2$$

l : index of feature layer in the perceptual network
λ_l : weight assigned to layer l
$\phi_l(\cdot)$: feature map extracted from layer l
x : ground-truth input image
$\hat{x}$: reconstructed / generated image
$\|\cdot\|_2^2$: squared ℓ_2 norm (Euclidean distance)

Perceptual loss in VQ-VAE encourages reconstructed images to match the original image in **feature space**, not only pixel space. Both the original and reconstructed images are passed through a pretrained perception network, such as VGG or LPIPS, and the loss is computed between selected intermediate feature maps.

This helps the decoder **preserve important perceptual features** such as edges, textures, shapes, and object-level patterns. Compared with pure pixel losses like L1 or MSE, which often average possible outputs and produce blurry reconstructions, perceptual loss encourages reconstructions that look closer to the original from a human visual perspective, even when exact pixel alignment is imperfect.

Modified Loss Funciton : Charbonnier Loss

$$\mathcal{L}_{\text{Charbonnier}} = \sum_i \sqrt{(x_i - \hat{x}_i)^2 + \epsilon^2}$$

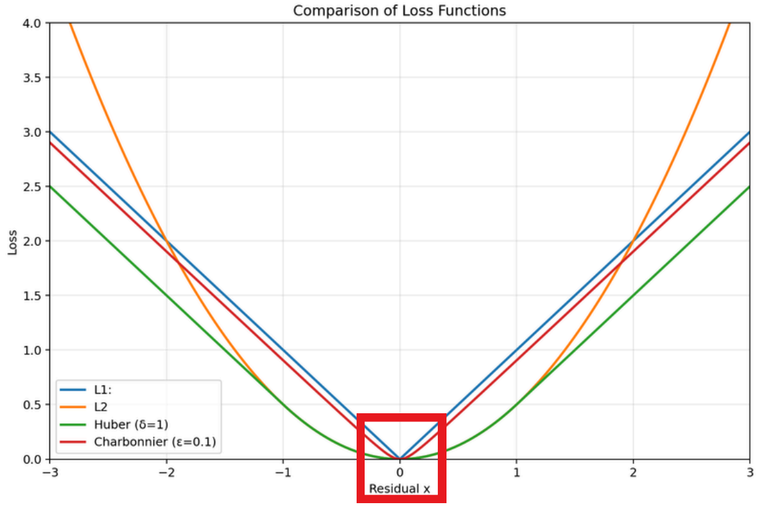
i : index of elements (e.g., pixels)

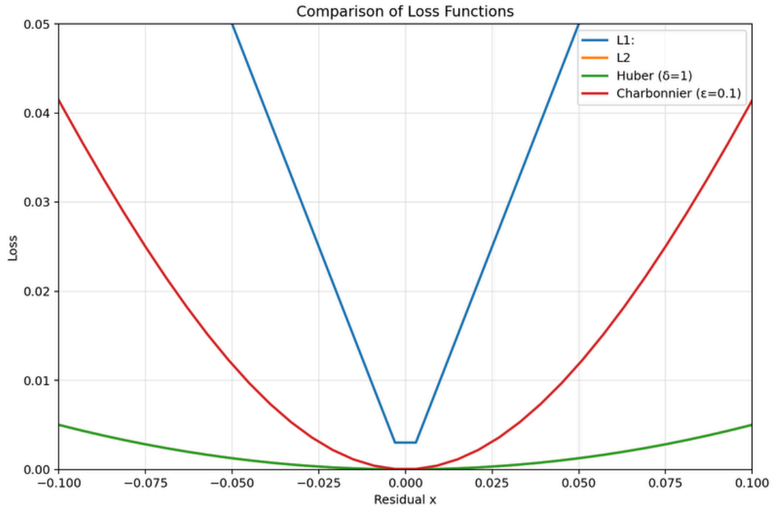
x_i : ground-truth value at position i

$\hat{x}_i$: predicted value at position i

ϵ : small constant for numerical stability (e.g., 10^{-3})

The Charbonnier loss is **superior** to Mean Squared Error in VQ-VAE's reconstruction loss because it **behaves like L1**, reducing sensitivity to outliers and preventing MSE's blur from over-penalizing large pixel errors. Unlike Huber which δ threshold tuning to decide where loss switches from L2-like to L1-like. Charbonnier also has ε but softer transition due to formulation.

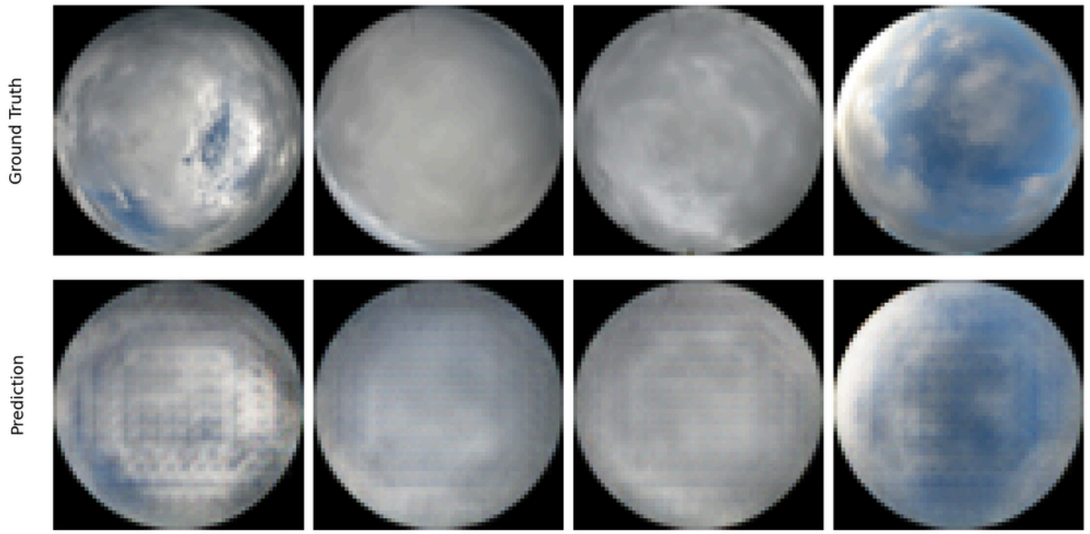




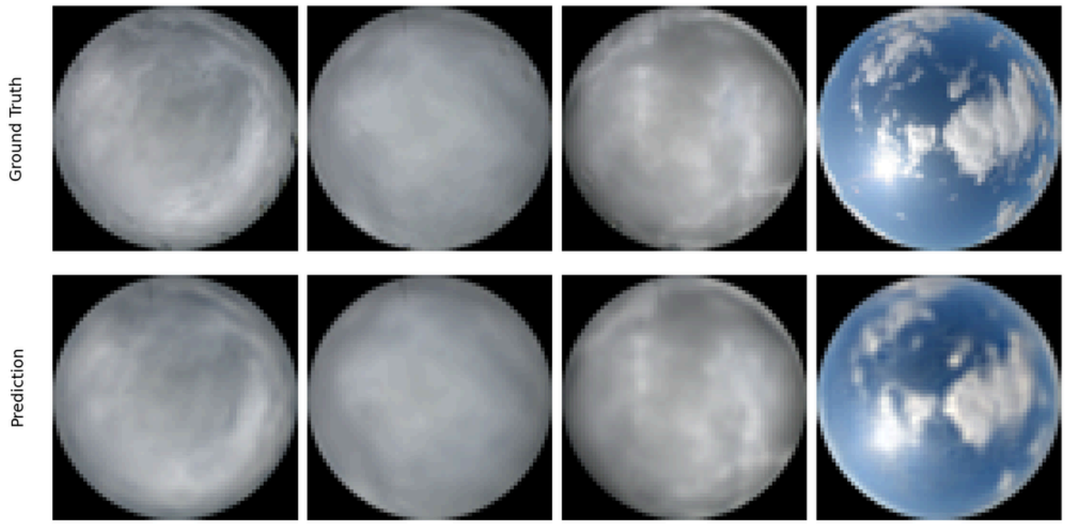
Interpolation Convolution Hashemi et al. (2019).

This design replaces strided transposed convolution with explicit upsampling in both temporal and spatial dimensions prior to convolution. This can improve smoothness and reduce undesirable interpolation artifacts. As a result, it improves reconstruction smoothness and helps mitigate checkerboard artifacts.

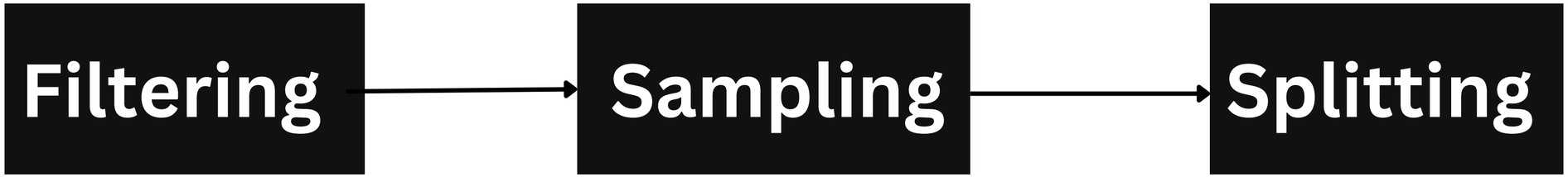
Without interpolation



With interpolation



Dataset



- **Filtering :** Nie et al.'s modified red-blue ratio to detect cloud pixels and **filter out** clear sky samples
- **Sampling :** Every 2 minutes (saves time, minimal loss in accuracy). Ensure 15 mins before & after have valid image + PV data
- **Splitting :** For Video Prediction Model → Input 8 images and target for 8 images
- **Data partitioning :** We have no fixed partitions since it's also included in model tuning

Dataset Analysis

Dataset	SKIPP'D	SIRTA
Resolution	64 × 64	64 × 64
Capture Interval	1 minute	1-2 minute
Start Date	09/03/2017	01/01/2023
End Date	26/10/2019	31/12/2023
Start Time	06:00	05:00
End Time	Not over 20:00	Not over 22:00

Table 3.1: Dataset properties

- SIRTA offers a shorter sky-image dataset: continuous measurements from over 100 sensors (clouds, aerosols, radiation, trace gases, etc.), collected across 2023, covering many meteorological parameters around Paris (solar irradiance , clear sky index)
- The dataset at the Stanford link is Sky Images and Photovoltaic Power Generation Dataset (SKIPP'D): it provides 2017–2019 fish-eye sky -images and matching PV power output every minute.
- We use these sky-image datasets to evaluate our **VQ-VAE** model, testing how well it compresses and reconstructs real atmospheric scenes in the implemented code.

New Sky Image Dataset at CUEE

- **Temporal Downsampling:** Uses OpenCV and conditional logic to extract frames at **even-minute intervals.**
- **Spatial Normalization:** Resizes all frames to a standard **1080x1080** square to reduce bitrate and size
- **Serialized Batch Processing:** Implements an idempotent pipeline and NumPy serialization within **Pickle files** to enable high-speed data loading and maximize efficiency.

By converting high-bitrate video into synchronized, resized image sets, this method achieved a 95% reduction in raw data—shrinking **400–500 GB** to just **10–20 GB** without sacrificing essential visual features.

Parameter tunings

Parameter	Sweep range	Best configuration
Codebook size (n_codes)	64, 128, 256, 512, 2048	2048
Residual layers (n_res_layers)	1–6	6
Hidden channels ($n\_hiddens$)	30–384	384
Embedding dimension	30–394	394
Downsample	$(2,2,2) - (4,4,4)$	$(2,4,4)$
Learning rate	10^{-6} to 10^{-1}	1×10^{-4}
Reconstruction weight ($\beta_{reconstruction}$)	1.0, 16.667	1.0
Perceptual weight ($\beta_{perceptual}$)	0.1–1	0.3
Commitment weight ($\beta_{commitment}$)	0.1–1.0	0.2
EMA decay	0.9–0.95	0.95
Restart threshold	0.9–1	1.0

- We tuned the VQ-VAE on the SIRTA sky-image datasets across architecture, training, and codebook settings.
- Performance generally improved with increased capacity or longer training, but further scaling was limited by available computational resources.

Best model by PSNR

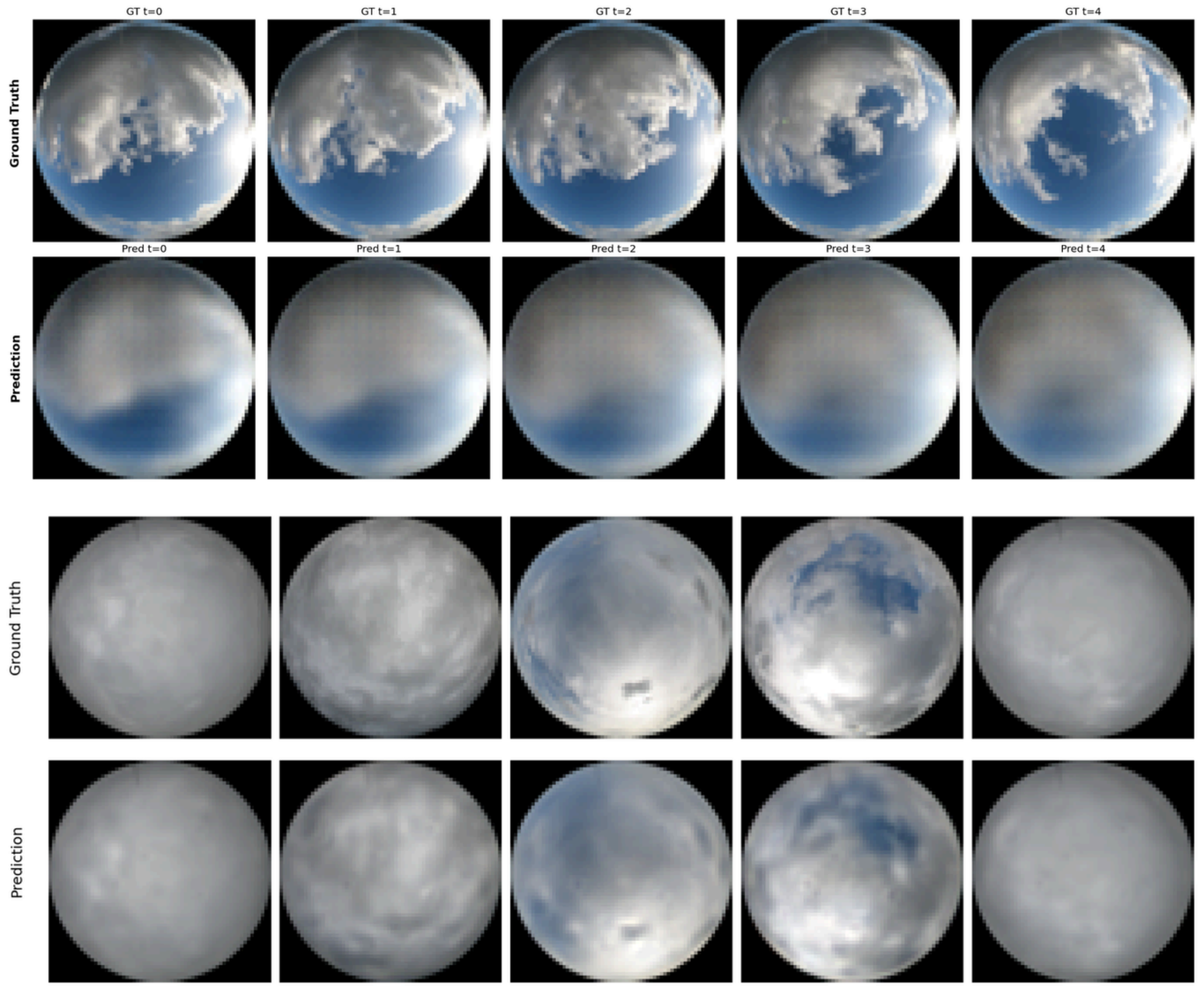
Rank	PSNR	Loss	codes	RL	hid	emb	β_{rec}	β_{com}
1	37.2580	0.013486	2048	6	384	394	1	0.20
2	37.0842	0.006936	2048	4	240	256	16.667	0.30
3	36.5880	0.030966	2048	4	240	256	4.000	0.30
4	35.9608	0.016386	2048	6	192	192	1.000	0.20
5	35.9160	0.009440	2048	4	240	256	16.667	0.30

The study concludes that VQ-VAE performance for cloud-motion reconstruction depends heavily on architectural capacity and stabilization techniques. By expanding the codebook to 2048 entries and employing deeper residual processing, the model captures fine-grained illumination and structural transitions that smaller baselines miss.

EMA-based updates, Charbonnier loss, and Interpolation Convolution mitigated earlier issues like "dead" codes and hardware-induced artifacts. Using the SIRTA dataset for tuning provided the necessary complexity to reach a high 37.32 dB PSNR. Ultimately, the results prove that discrete latent modeling is highly effective when optimized for expressivity and training stability.

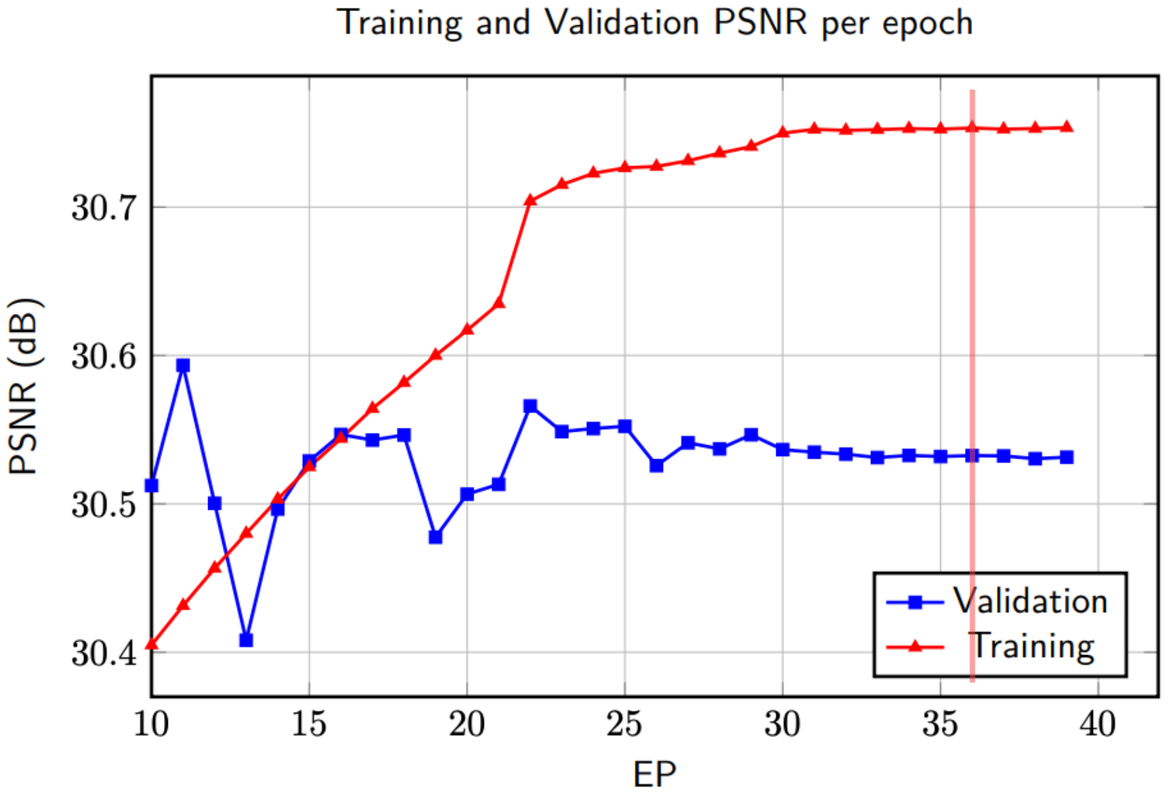
Original Implemetation

Ours



SkyGPT Training Dynamics

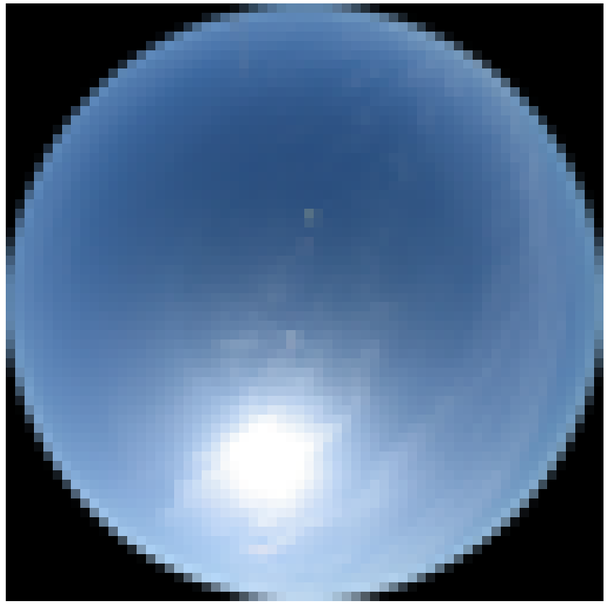
Following the implementation of a VQVAE model for high accuracy image encoding, the generative SkyGPT model was trained to establish a performance benchmark.



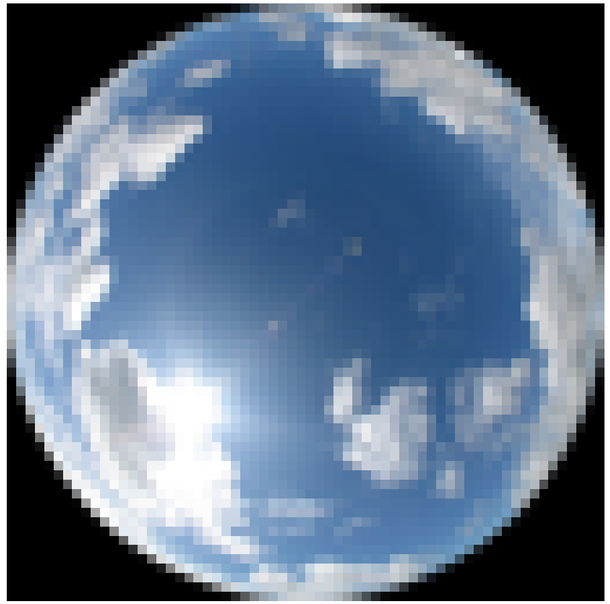
- **Training PSNR:** Demonstrates a consistent upward trajectory with a distinct performance shift at Epoch 22, ultimately converging at a maximum value of 30.73 dB.
- **Validation PSNR:** Reflects the stochastic nature of complex cloud modeling through initial volatility before stabilizing at 30.53 dB after Epoch 30.
- **Stabilized Performance:** Reaches peak predictive performance at Epoch 36, with results remaining steady throughout the final training stages.

SkyGPT Reconstruction Performance

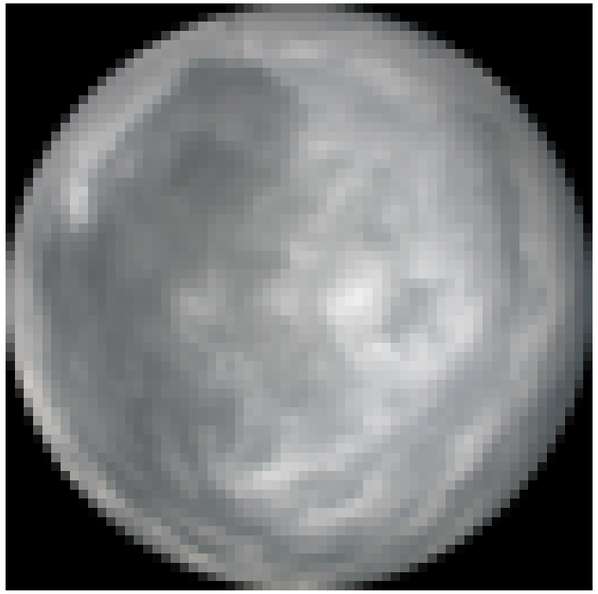
Validation results at Epoch 36 demonstrate SkyGPT's performance across three distinct sky scenarios, highlighting its reconstruction accuracy in diverse weather conditions.



Clear Sky Conditions



Partly Cloudy and
Dynamic Cloud Motion

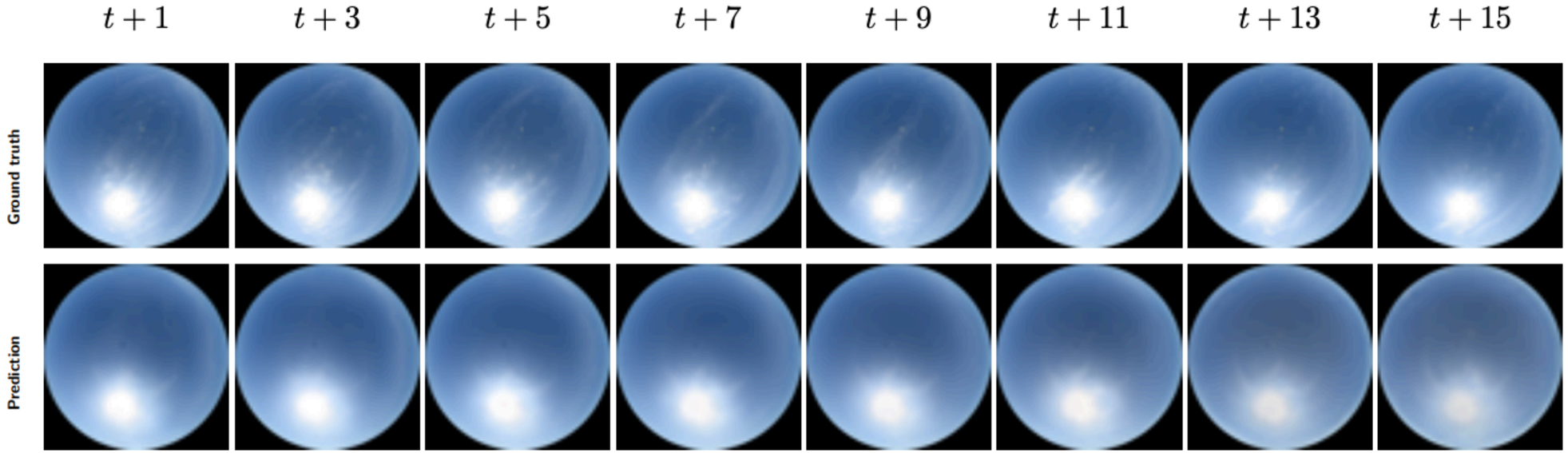


Overcast and Dense
Cloud Conditions

Clear Sky Conditions

The model's precise positioning of the primary irradiance source ensures a reliable and temporally consistent reconstruction for stable atmospheric conditions.

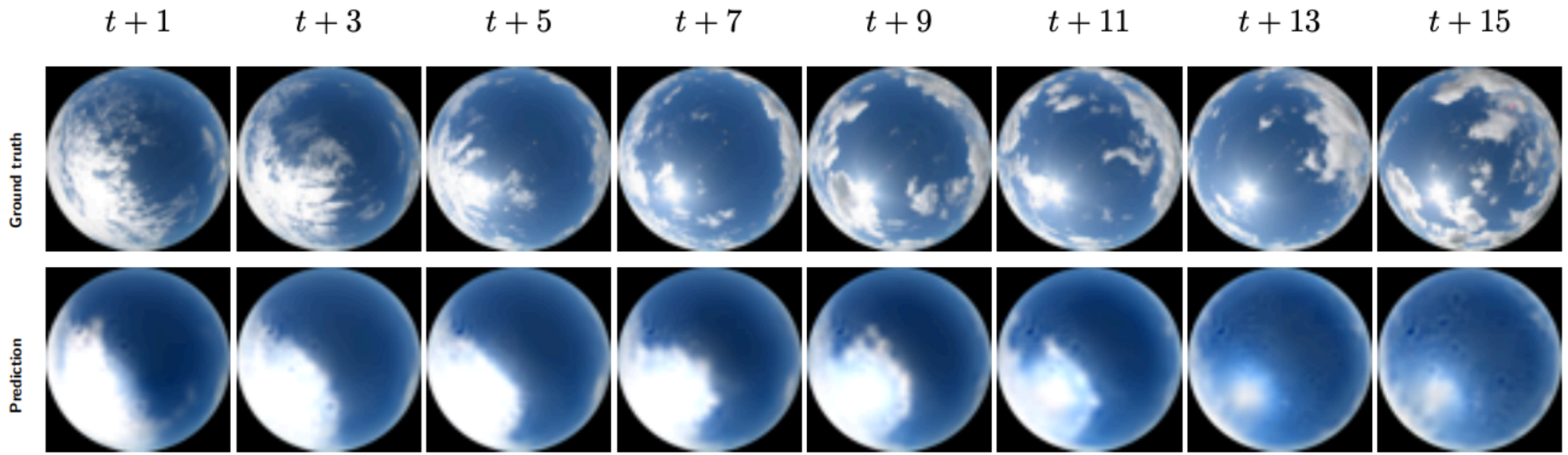
- **Solar Trajectory Tracking:** Accurately captures sun movement across zenith and azimuth angles, preventing spatial drifting throughout the prediction.
- **Stability & Logic:** Maintains a smooth, flicker-free sky gradient by correctly identifying the absence of cloud motion.



Partly Cloudy and Dynamic Cloud Motion

By combining sharp visual reconstruction with accurate motion extrapolation, SkyGPT provides a robust framework for predicting high-contrast sky conditions and supporting solar energy forecasting.

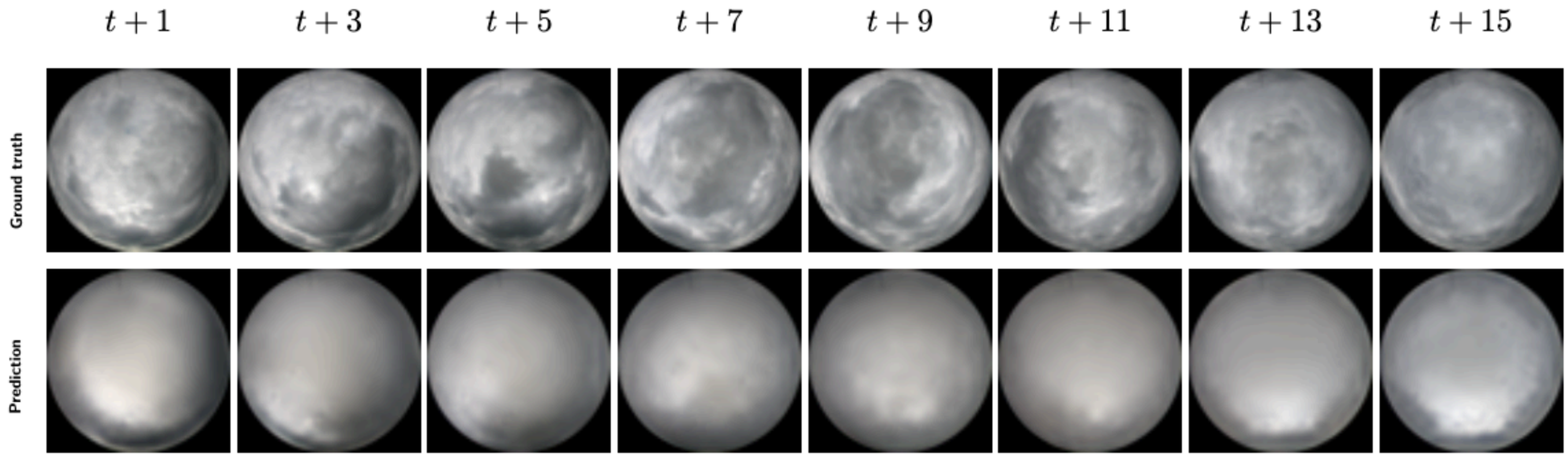
- **Dynamic Capture:** Captures continuous deformation, condensation, and evaporation of discrete clouds as they traverse the field of view.
- **Accurate Motion Trends:** Extrapolates general cloud movement trends while accounting for inherent stochasticity to match ground truth.



Overcast and Dense Cloud Conditions

While the model effectively captures general light levels, these results highlight the difficulty of reproducing complex structural details within the cloud ceiling under low-contrast conditions.

- **Reduced Visual Fidelity:** The model struggles to preserve fine spatial details, leading to predicted images that appear significantly blurrier than the source.
- **Averaging of Stochastic Textures:** In low-contrast environments, the system tends to average out plausible future states, producing a "smeared" visual output.



Thank you!

During the preparation of this work, ChatGPT has been used solely for enhancing the readability and language.

After using this tool, we have reviewed and edited the content as needed and take the full responsibility for the content.

Mr. Tiranut Kanjanatunyarut
Mr. Wachirawit Piyaprapapan