Motivation

**Problem**

*Conditional Diffusion Models (CDM)*

1. Types of Conditions: labels, text, segmentation, maps, sketches, landmarks, face IDs, style images, etc
2. 2 types of CDMS – ways to add conditional information into diffusion models 🡪  
   **training-required CDMs** and **training-free CDMs**
   1. **(1) Training-required CDMs**
      1. Most CDMs use **training-required classifiers** to enforce and **add conditionality** to their models.   
         2 types of **training-required classifiers** 
         1. Time-dependent classifier
         2. Condition-dependent score estimator
   2. **(2) Training-free CDMs**
      1. Training-free CDMs try to also feed conditional information to the main model without extra training.
      2. It uses cross-attention control to realize the conditional generation, directly modifying the intermediate results during sampling to achieve zero-shot image restoration
3. Popular existing models: Stable Diffusion, ControlNet, …

*Current Problems*

1. Problem with **Training-required CDMs** (**training-required classifiers)**

They arenot flexible. Once a new target condition is needed for generation, they have to retrain or finetune the models, which is inconvenient and expensive.

2. Problem with **Training-free CDMs**

While they are effective in a single application, they are difficult to generalize to a wider range of conditions, e.g., style, face ID, and segmentation masks.

Overview of Free conditional Diffusion Model (FreeDoM)

**Introduction**

Free conditional Diffusion Model (FreeDoM) can be used for various conditions.   
Specifically, leveraging off-the-shelf pretrained networks, such as (Stable Diffusion, ControlNet), to construct time-independent energy functions, which guide the generation process without requiring training.

1. Firstly, to emphasize generalization, FreeDoM propose a sampling process guided by the energy function
   1. The energy function is very flexible to construct and can be applied to various conditions
2. Secondly, to make the proposed method training-free, FreeDoM uses off-the shelf pre-trained time-independent models, which are easily accessible online, to construct the energy function

FreeDoM

**Preliminaries (Sampling)**

*Understanding how condition information is added to generative models (here, it is added during sampling with the corrective score gradient that is calculated 2 ways – Training-required classifiers / Training-Free Methods)*

1. Score-based Diffusion Models (SBDM) (Unconditional)

In-essence, Unconditional SBDM estimate the score function  for denoising.

Sampling (denoising), to get Xt-1 from Xt, has the formula:

, where the model, estimated score s(Xt, t) is 

2. Conditional Score Function

Conditional sampling (denoising), to get Xt-1 from Xt, now includes conditional information to aid sampling. We modify the score in SBDM  to include conditional information for the direction as  Using the Bayesian formula , we can simplify 

. Subbing into the SBDM unconditional model, we get 

The new additional term  is the correction gradient that adds conditional information, pointing to a hyperplane in the data space, where all data

are compatible with the given condition.

**Training-required CDMs** introduces **training-required classifiers** like time-dependent classifiers to compute the correction gradient for conditional guidance.

3. Energy Diffusion Guidance

Alternative to training-required classifiers, to model the correction gradient , we can use an energy function:

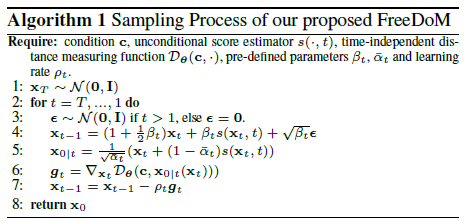
, where lambda = positive temperature coefficient, Z = normalising constant, E(c, Xt) is an energy function that measures the compatibility between the condition c and the noisy image Xt   
Energy function condition: any function that obeys 🡪 the smaller the value, the more compatible Xt is with c (energy function = 0 if Xt is perfectly compatible with c).

We sub in the energy function expression to get the following:

  
New conditional sampling function with energy function ,  
 where ,  = learning rate of the correction term.

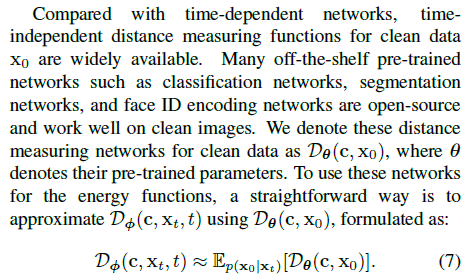
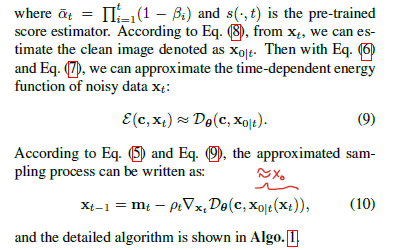
**FreeDoM**

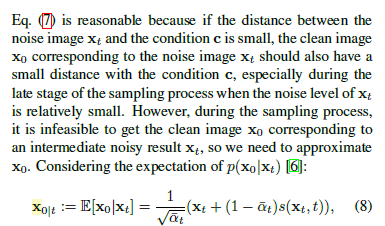
1. Approximate Time-Dependent Energy (chain of approximations)



Energy functions are time-dependent  .

Existing classifier-based methods (e.g., Time-dependent Classifiers), choose time-dependent distance measuring functions  to approximate , where  computes the distance between the given condition c and noisy intermediate results xt, and  defines pre-trained parameters.  
FreeDoM uses time-independent distance measuring functions, making the method training-free and flexible for various conditions.

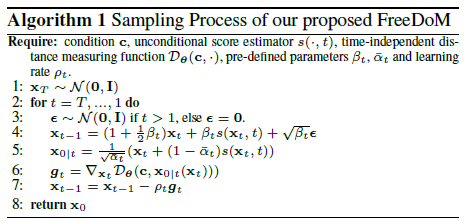


This is where the magic happens. From the different conditions and a pre-trained model used for each respective conditions, we are able to obtain the quantity , an approximation of the energy function  used in the sampling formula , that adds conditional information during sampling (inference time).

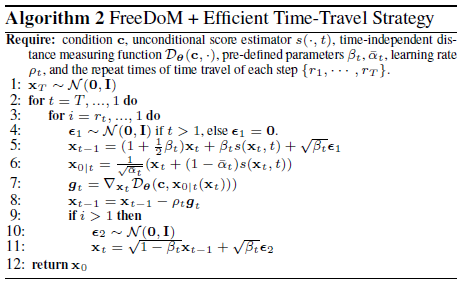
This allows condition information added in inference time to guide image generation based on conditions instead of adding conditional information in training time (without computationally expensive training-required classifiers that add conditional information)

2. Efficient Time-Travel Strategy

*Original*



*Efficient Time-Travel Strategy*





The time-travel strategy is a technique that takes the current intermediate result Xt back by j steps to X(t+j), then resamples it to the t-th time step again.

This strategy inserts for sampling steps into the sampling process and refines the generated results.   
**In the paper j = 1, we go back by 1 step each time and resample.**

This resampling process is repeated by rt times at the t-th time step.

We only do this in the small range of time steps, the middle time steps (the semantic stage), for the change in the generated result to be significant with respect to the trade-off of increased time complexity due to more sampling steps.

**The range of the semantic stage we do this resampling (time-travel strategy) is an experimental choice depending on the specific diffusion models we choose.**

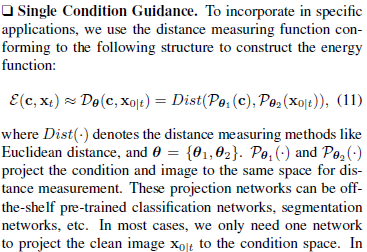
In Algo 2, we see that when rt = 1 means we do not apply the time travel strategy in the t-th time step.

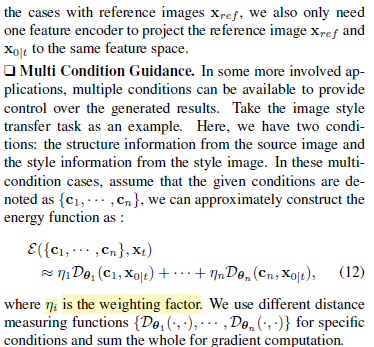
3. Construction of the Energy Function

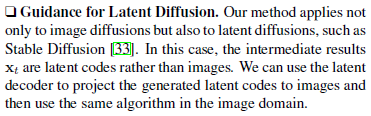
Constructing the term , an approximation of the energy function  used in the sampling formula , that adds conditional information during sampling (inference time).

3 cases:

1. adding single conditions guidance
2. adding multiple conditions guidance
3. Guidance for latent diffusion (not just images)





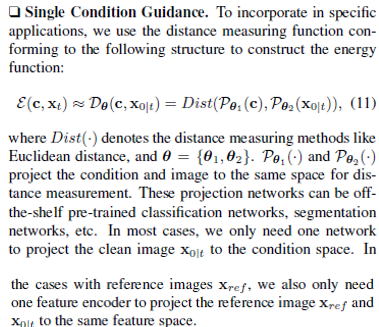
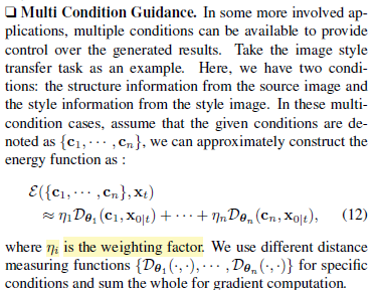


4. Specific supported conditions **(IMPLEMENTATION)**

*Recap:*

We need to construct the term , an approximation of the energy function  used in the sampling formula , that adds conditional information during sampling (inference time).

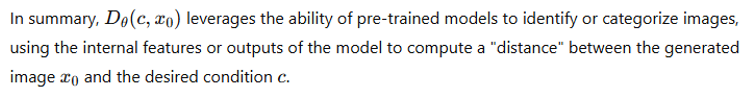
 is calculated as such:

 can be calculated as such .

It is dependent on the specific condition, that controls:

1. A chosen distance function Dist()
2. A chosen condition specific pre-trained model, theta.
   1. In most cases, we do not need to split theta into theta1 and theta 2, and only need one network to project the clean image x0 the condition space.
   2. Most cases, P(c) = model output for given condition (eg a given segmentation map) and P(X0 | t) is the model output for the approximated clean input X0 | t with the derived formula above 



*Supported Conditions  
(Derivation of*  *for each supported condition is different)*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Condition | Model |  |  | Distance function, Dist() |
| Text | CLIP | CLIP text encoder (given text) | CLIP image encoder () | L2 distance for |
| Segmentation Maps | BiSeNet (Face Parsing Network) | Given parsing map | BiSeNet () | L2 distance for |
| Sketches | Open-source pre-trained network that transfers a given anime image to the style of hand-drawn sketches.  Adversarial Open Domain Adaptation for Sketch-to-Photo Synthesis  Xiaoyu Xiang, Ding Liu, Xiao Yang, Yiheng Zhu, Xiaohui Shen, Jan P. Allebach | Given Sketches | Model () | L2 distance for |
| Landmarks | Open-source pre-trained human  face landmark detection network.  First stage finds  the position of the centre of a face.  Second stage  marks the landmarks of this detected face.  Cunjian Chen. PyTorch Face Landmark: A fast and accurate  facial landmark detector, 2021. | Given Landmarks Conditions | Model () | L2 distance for  \* Only use the  gradient in the face area detected in the first stage to update the intermediate results. |
| Face IDs | ArcFace  (Open-source pre  trained  human face recognition network, to extract the target features of reference faces to represent face IDs) | Model (Reference Image)  = Extracted ID features of reference image | Model ()  = Extracted ID features of | L2 distance for |
| Style Images | CLIP | ,  See distance cell for the meaning of | ,  See distance cell for the meaning of | (Squared) L2 distance    Where   is the style image,  denotes the Gram matrix of the j-th layer feature map of the image encoder  In our experiments, we  choose the features from the third layer of the CLIP image  encoder to generate satisfactory results. |
| Low Pass Filters | For the image transferring task, we  need an energy function to constrain the generated results  conforming to the structure information of the source image .  Similar to EGSDE [50] and ILVR [5], we choose a  low-pass filter  for this setup. |  |  | (Squared) L2 distance for |

5. Experimented Models **(IMPLEMENTATION)**

- Along with the specific supported conditions (4), using the different pre-trained models for finding the  that approximates the energy function that is used in the correction gradient  that supplements conditional information,

- We need the already pre-trained generative diffusion models (not to be confused with the pre-trained models for finding  in (4).

- These are the generative models (without the new conditions) that will be tested with the new conditions (training-free Freedom algorithm), without any new training or fine-tuning to include new conditional information (basically the essence of FreeDoM – training-free method for adding new conditions for generation)

**- DDIM + 100 steps is chosen as the sampling strategy**

|  |  |
| --- | --- |
| **Model** | **New Conditions** |
| Unconditional Human Face Diffusion Model   * Pre-trained on CelebA-HQ dataset | Text  Parsing maps Sketches  Landmarks  Face IDs |
| Unconditional ImageNet Diffusion Model   * Pre-trained on ImageNet dataset | Text  Style |
| Conditional (Time-dependent classifier-based) ImageNet Diffusion Model   * Pre-trained on ImageNet dataset | Style |
| Stable Diffusion   * Pre-trained pose-to-image * Pre-trained scribble-to-image | Style |
| ControlNet  (Stable Diffusion based with extra conditions) | Face IDs  Style |

Notes:

* For latent diffusion models, Stable Diffusion and ControlNet, needed to add the training-free conditional interfaces (just asking user for conditions, inbuild = text prompt interface) based on their energy functions to work with the existing training-required conditional interface
* Smaller diversity datasets (Human Face Images) do not require time-travel strategy,  
  Larger diversity datasets (ImageNet, 1000 classes) require time-travel strategy for stronger guidance for better results
* Multiple conditions tested on:

1. Unconditional Human Face Diffusion Model
   1. {Face ID + Landmarks}, {Parsing Map + Text Prompt}
2. Unconditional ImageNet Diffusion Model
   1. {Style + Text Prompt}, {Style + Source Image}

* The energy function’s learning rate,ρt, in  is also an adjustable hyperparameter, users can adjust the intensity of control as needed

6. Limitations

1. FreeDoM’s sampling time cost of still higher than the training-required methods because
   1. each iteration adds a derivative operation for the energy function, and
   2. time-travel strategy introduces more sampling steps
2. Difficult to use the energy function to control the fine-grained structure features in the large data domain.
   1. For example, using the Canny edge maps as the conditions may result in poor guidance, even if we use the time-travel strategy.
   2. In this case, the training-required methods will provide a better alternative.
3. Multi-condition control difficulties
   1. A math equations on a white background

      Description automatically generated deals with multi-conditional control guidance and assumes that the provided conditions are independent, which is not necessarily true in practice.
   2. When conditions conflict with each other, FreeDoM may produce subpar generation results.

**Summary of FreeDoM**

1. Conditional information can be added to generative models in numerous ways (addition of conditional information via the corrective gradient score )
   1. 1. Training-Required and 2. Training-Free
   2. Conditional Score-based Diffusion Model Sampling:
   3. We can use energy functions to approximate the corrective gradient score  
      
2. FreeDoM is a training-free energy-guided conditional diffusion model, addressing a wide range of conditional generation tasks without training.
   1. The method uses off-the-shelf pre-trained time-independent networks to approximate the time-dependent energy functions with time-independent approximated distances between intermediate results and the condition.
      1. Single Condition: 
      2. Multiple Condition: A white background with black symbols

         Description automatically generated
      3. Each single condition, a different formula 
         1. Dependent on the condition (Chosen Pre-trained Model for the condition specific task to find the distance (“closeness” of approximated intermediate result and the condition)
         2. Distance formula (e.g. L2...)
         3. See (4) for supported conditions
   2. Need extra Time-travel strategy to combat poor guidance for datasets with large number of classes
   3. Then, using the gradient of the approximated energy to guide the generation process.  
      Sampling (Denoising Step: )
3. FreeDoM supports different diffusion models, including image and latent diffusion models (Stable Diffusion, ControlNet)
   1. **The range of the semantic stage we do this resampling (time-travel strategy) is an experimental choice depending on the specific diffusion models we choose.**

**Training Setup of FreeDoM**

**A maths and equations on a white background

Description automatically generated with medium confidence**

*Fixed*

1. j, time-travel step: takes the current intermediate result Xt back by j steps to X(t+j), then resamples it to the t-th time step again, = 1

2. Sampling strategy = DDIM + 100 time steps

*Dynamic*

1. The energy function’s learning rate,ρt, in  is also an

2.  is **dependent on the specific supported condition see (FreeDoM – (4))**

3. {t, for t time step we do time-travel}: the range of the semantic stage we do this resampling (time-travel strategy) = an experimental choice **depending on the specific diffusion models we choose.**

Code Analysis

**(Face GD Folder)**

**Experiment : Unconditional Human Face Diffusion Model**

* Pre-trained on CelebA-HQ dataset
* Implemented Training-Free Guided Conditions:
  + Text
  + Parsing maps Sketches
  + Landmarks
  + Face IDs

*In main.py, main(): runner.sample(args.sample\_strategy)*

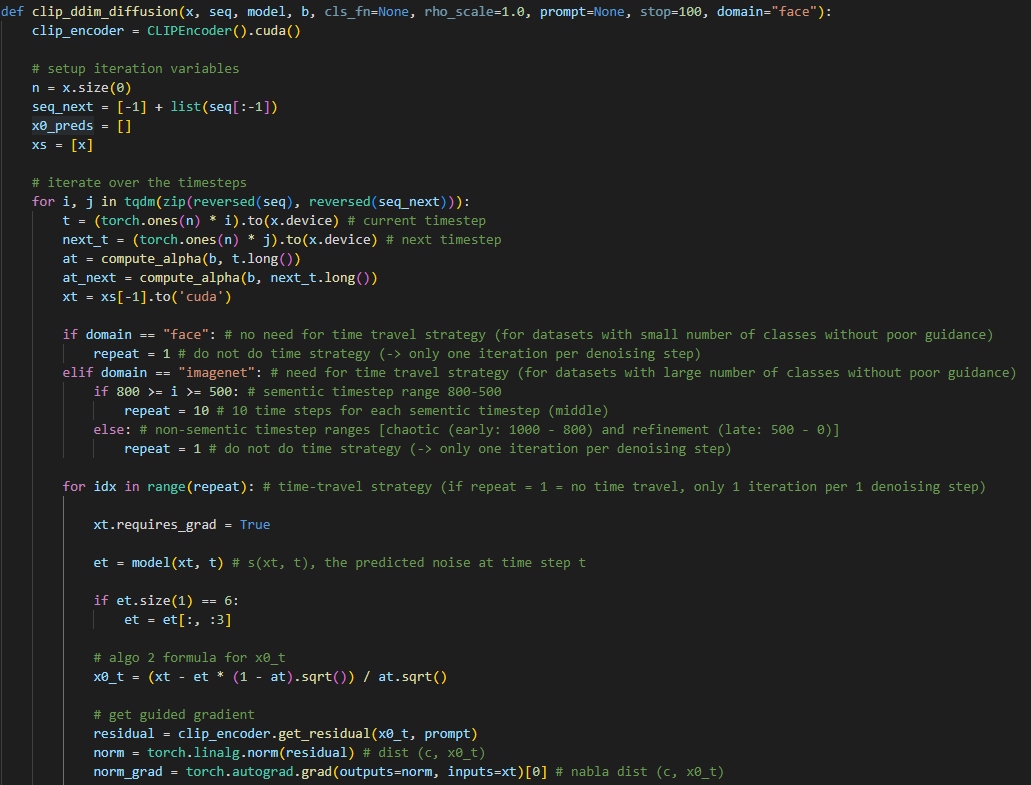
* Pass in "-s" of command line argument (type of condition)

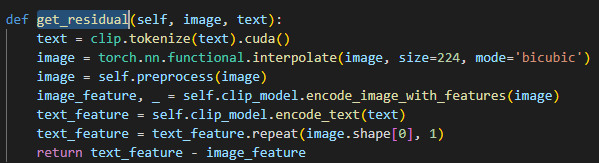
*Goes to runner.py, class Diffusion (sequentially downwards):*

1. *sample(self, mode)*
   1. Initialize the diffusion model (whether imagenet or celeba face model)
   2. Calls sample\_sequence(model, cls\_fn, mode), mode = args.sample\_strategy
2. sample\_sequence(model, cls\_fn, mode)
   1. randomly initializes a noise vector (max noise), size = (1, 3, 256, 256)
   2. for the appropriate condition (sampling strategy), e.g. “clip\_ddim” (text condition), go to the corresponding condition denoising function e.g. “sample\_image\_alogrithm\_clip\_ddim()” get the denoised image
   3. From above, called the specific condition sample function “sample\_image\_alogrithm\_XXXXX\_ddim()”
3. sample\_image\_alogrithm\_XXXXX\_ddim():
   1. number of sequence same for all
   2. calls the specific condition denoising function:  
      XXXXX\_ddim\_diffusion() in */function/denoising.py*

**Clip (Text Prompt Condition):**

*Goes to /function/denoising.py: clip\_ddim\_diffusion() 🡪 Algorithm 2 for Clip*



****

**Clip (Text Prompt Condition):**

*Goes to /function/denoising.py: clip\_ddim\_diffusion() 🡪 Algorithm 2 for Clip*

Appendix

Glossary

1. Different Conditions for Facial Image Generation:  
     
   In the context of diffusion models for image generation, this statement means that the generation process is conditioned by providing specific inputs (segmentation map, sketch, landmarks, and face ID) that guide how the model generates or reconstructs images. Here’s how each component could work to influence the generated output:

Segmentation Map: This map specifies regions or parts of the image, helping the model understand the structure and boundaries of different parts, such as separating a face from the background. It guides the model to fill in details based on the locations and boundaries defined.

Sketch: A sketch provides an **outline or simplified version** of the intended shape and structure. This can act as a basic blueprint, letting the model add texture and finer details to match the input sketch.

Landmarks: For faces, landmarks typically indicate the **positions** of key facial features (e.g., eyes, nose, mouth). These points help the model place facial features correctly, ensuring that the generated image adheres to the desired facial structure.

Face ID: Face ID could involve providing a **specific facial identity or characteristics** the model should mimic. This could guide the generation process to align with particular facial attributes associated with a person’s identity.

Using these conditions in an otherwise unconditional diffusion model allows the model to generate images that align with the desired structure, appearance, and identity specified by the conditions, even though the original diffusion model may not be inherently designed to generate such specific outputs.

1. Time-Dependent Classifiers in Conditional Diffusion Models:

In diffusion models, "time" typically represents the noise level, with early stages having less noise (closer to the original image) and later stages having more noise (closer to complete random noise). A time-dependent classifier is trained to predict certain attributes or classes of data at each noise level (or "time step") to guide the generation process.

A time-dependent classifier is an auxiliary model used to guide a conditional diffusion process by predicting or enforcing certain conditions at each noise level (time step).

In conditional DDPMs with time-dependent classifiers, the classifier may learn to predict class labels or other attributes across different noise levels. The diffusion model then uses this classifier’s feedback to adjust generation at each step to meet the condition (e.g., creating a dog image rather than a cat).

In some advanced conditional DDPMs, time-dependent classifiers are added to guide the model toward generating images that meet specific conditions. However, this is an additional training step that requires both the diffusion model and the classifier to be trained together (or fine-tuned), which adds complexity and computational cost. This type of conditional setup is not part of traditional DDPMs but is rather a modification for specific conditional generation tasks.

1. Condition-dependent score estimators in Conditional Diffusion Models:

Condition-dependent score estimators are models used in conditional diffusion frameworks to estimate the gradients (or “scores”) that guide the generation process toward specific conditions. In a diffusion model, a score estimator predicts the noise direction at each time step, helping the model denoise the image gradually to produce a final output. When this score estimation is conditioned on specific information (such as class labels, sketches, or text prompts), it becomes a condition-dependent score estimator.

In a conditional diffusion model, the generation process is steered by conditions (e.g., class labels or other input information) to produce images that satisfy these conditions. The score estimator is trained to incorporate these conditions into its predictions, meaning that at each noise level or time step, it considers both the noisy image and the condition to make a better noise prediction.

For example, if the condition is a class label "dog," the score estimator will learn to denoise in a way that the final image aligns with the appearance of a dog. This conditioning information influences the score estimation at every step, guiding the model to generate images consistent with the given condition.

To incorporate condition dependency, score estimators are often trained with both the noisy image and the condition as inputs, learning to predict the noise component specifically for that condition. This makes them condition-dependent and increases training costs, as the model must learn not only to denoise but to do so according to a variety of conditions.

1. Cross-Attention Control for Zero-Shot Conditional Generation:

Cross-attention is a mechanism in which a model learns to focus on specific parts of the input when generating each part of the output, which is useful in achieving conditional generation. In training-free CDMs, cross-attention control can be applied to directly influence intermediate results during diffusion without requiring additional training.

In this context, cross-attention control allows the model to “focus” on specific conditional features (such as attributes, objects, or colours specified in the prompt or guidance) while it generates the image. This control is applied as the model goes through its iterative steps, modifying the generated content to meet the desired conditions.

Because this is applied at the intermediate stages without retraining, the model can perform zero-shot image restoration—generating images that meet specific conditions on the fly, even if it has not been explicitly trained for that condition. This approach effectively reorients the model’s focus at each step, guiding the output without extra training.

1. DDIM (Denoising Diffusion Implicit Models)

A screenshot of a white text

Description automatically generated

