# DITTO: Diffusion Inference-Time $T$-Optimization for Music Generation

*Zachary Novack[1,2], Julian McAuley[1], Taylor Berg-Kirkpatrick[1], Nicholas J. Bryan[2]*

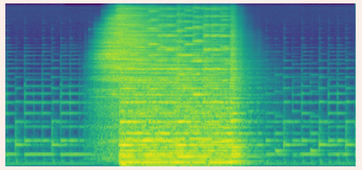
[1]UC – San Diego [2]Adobe Research

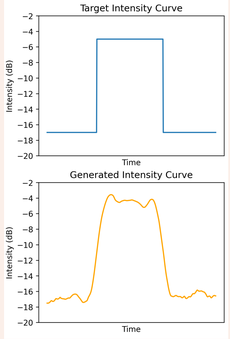
Project Personal

## Control Tasks

### Intensity Control

$f(\cdot)$ = Smooth RMS Energy, $\mathcal{L}$ = MSE

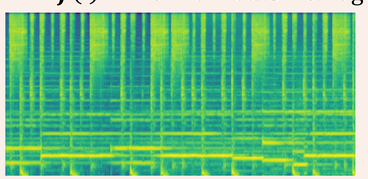


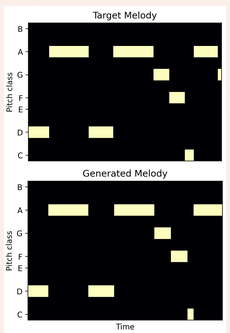



| Control / Metric | Intensity MSE (↓) | Intensity FAD (↓) | Intensity CLAP (↑) |
|---|---|---|---|
| Default TTM | 40.843 | 0.707 | 0.373 |
| ControlNet | 38.411 | **0.637** | 0.308 |
| FreeDoM | 23.292 | 0.673 | **0.482** |
| DOODL | 4.785 | 0.695 | 0.342 |
| DITTO (ours) | **4.758** | 0.682 | 0.433 |

### Melody Control

$f(\cdot)$ = Normalized Chromagram, $\mathcal{L}$ = CELoss

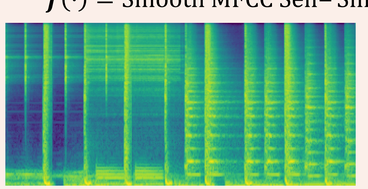


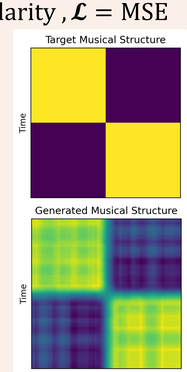



| Control / Metric | Melody Acc (↑) | Melody FAD (↓) | Melody CLAP (↑) |
|---|---|---|---|
| Default TTM | 10.527 | 0.707 | 0.373 |
| ControlNet | 81.353 | **0.545** | **0.478** |
| FreeDoM | 31.544 | 0.706 | 0.477 |
| DOODL | 81.592 | 0.715 | 0.336 |
| DITTO (ours) | **82.625** | 0.699 | 0.432 |

### Structure Control

$f(\cdot)$ = Smooth MFCC Self– Similarity , $\mathcal{L}$ = MSE

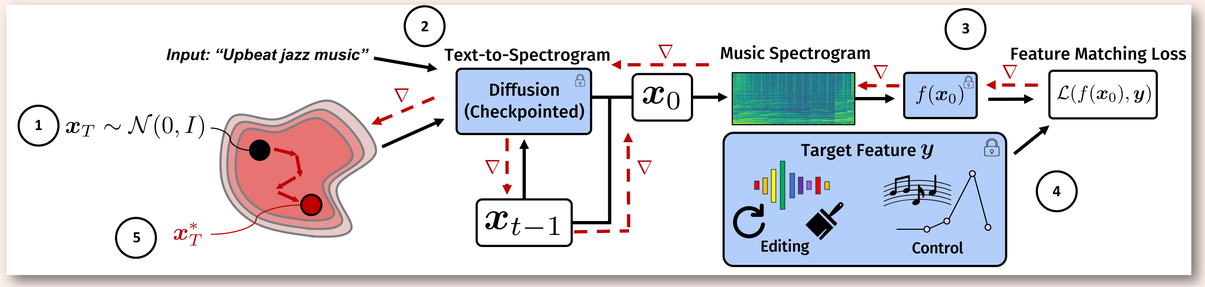





| Control / Metric | Structure MSE (↓) | Structure FAD (↓) | Structure CLAP (↑) |
|---|---|---|---|
| Default TTM | 0.309 | 0.707 | 0.373 |
| ControlNet | – | – | – |
| FreeDoM | **0.018** | 0.668 | 0.415 |
| DOODL | 0.074 | 0.653 | 0.387 |
| DITTO (ours) | 0.024 | **0.632** | **0.418** |

## DITTO: Diffusion Inference-Time T-Optimization

***"writing about music is like dancing about architecture"***




**TLDR:** We optimize the **initial noise latent $x_T$** through diffusion sampling to match some feature of our generation $f(x_0)$ to the target feature $y$, w/**gradient checkpointing** for efficient memory usage

## Method Comparison

### Training-Based (Conditioning)

Music-ControlNet, JASCO

✓Arbitrary controls
✓Control/quality balance
✓Fast @ inference time

⊗Large-scale training
⊗Paired/labeled control data
⊗Fixed controls @ training

### DITTO (Optimization)

✓Any (differentiable) control
✓Architecture/sampler agnostic
✓Zero training
✓Exact control gradients

⊗Slow @ inference time

### Training-Free (Guidance)

Classifier Guidance, FreeDoM

✓Any (differentiable) control
✓Zero training
✓Moderate inference costs

⊗Approximate gradients
⊗Limited control in low SNR
⊗Bad at fine-grained controls

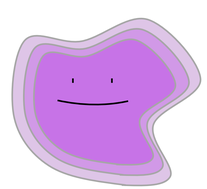
## And More!

**Further Analysis:**

- Subject listening study
- Efficiency evaluation w/DOODL

**Extra Use Cases:**

- Correlation-based intensity control
- Multi-feature optimization
- Reference-free looping
- Musical structure transfer
- Optimized latent reuse
- Real-audio inversion



## Editing Tasks

### Outpainting

$f(\cdot)$ = 1 (LR)Pixel Mask, $\mathcal{L}$ = MSE

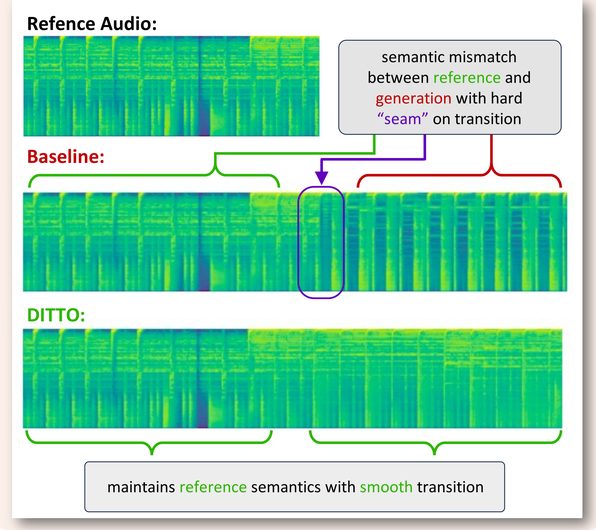
### Inpainting

$f(\cdot)$ = 2 (LL, RR) Pixel Masks, $\mathcal{L}$ = MSE

### Looping

$f(\cdot)$ = 2 (LR, RL) Pixel Masks, $\mathcal{L}$ = MSE

| Method | Outpainting $o=1$ | Outpainting $o=2$ | Outpainting $o=3$ | Inpainting $o=2$ | Inpainting $o=3$ | Inpainting $o=4$ | Looping $o=1$ |
|---|---|---|---|---|---|---|---|
| DOODL | 0.719 | 0.707 | 0.700 | 0.696 | 0.693 | 0.688 | 0.750 |
| Naive | 0.722 | 0.716 | 0.712 | 0.707 | 0.705 | 0.697 | 0.753 |
| MD | 0.733 | 0.716 | 0.710 | 0.701 | 0.694 | 0.690 | 0.749 |
| MD-50 | 0.718 | 0.714 | 0.705 | 0.711 | 0.708 | 0.701 | 0.752 |
| GG | 0.754 | 0.738 | 0.719 | 0.717 | 0.709 | 0.700 | 0.774 |
| FreeDoM | 0.726 | 0.723 | 0.715 | 0.719 | 0.709 | 0.704 | 0.758 |
| DITTO (ours) | **0.716** | **0.703** | **0.698** | **0.690** | **0.688** | **0.686** | **0.746** |




## Algorithm

**Algorithm 1** Diffusion Inference-Time $T$-Optimization (DITTO)

**input** : $\epsilon_\theta$, `Sampler`, sampling steps $T$, feature extractor $f$, loss $\mathcal{L}$, target feature $y$, starting latent $x_T$, text conditioning $c$, optimization steps $K$, optimizer $g$.

// Run optimization
**for** $i = 1$ to $K$ **do**
  $x_t \leftarrow x_T$ // Initialize noise latents
  **for** $t = T$ to 1 **do** // Diffusion sampling w/checkpointing
    $x_{t-1} = \texttt{Checkpoint}(\texttt{Sampler}, \epsilon_\theta, x_t, t, c)$
  **end for**
  $\hat{y} = f(x_0)$ // Extract features from generated output
  $x_T \leftarrow x_T - g(\nabla_{x_T} \mathcal{L}(\hat{y}, y))$ // Compute feature loss & backprop
**end for**

**output** : $x_0$