# ATOM-TTS: A LIGHTWEIGHT ON-DEVICE TTS SYSTEM ADOPTING CASCADED PREDICTION AND FRAME-DRIVEN VOCODER

**Anonymous authors**
Paper under double-blind review

## 1 ABSTRACT

On-device speech synthesis needs to balance speech quality under a low computing budget. However, existing methods struggle to model high-quality intermediate audio representations with a small number of parameters. Moreover, high-quality vocoders incur substantial inference overhead on edge devices with constrained computing resources. To address these issues, we propose Atom-TTS. Starting from acoustic prediction and waveform reconstruction, it reduces inference costs while maintaining synthesis quality. First, we propose the Cascaded Parameter Acoustic Predictor (CPAP), which adopts a TCN encoder for cascaded encoding and establishes explicit conditional relationships among parameters to efficiently predict intermediate representations. It compresses trainable parameters to 161k without degrading prediction quality. Next, we propose the Frame-Driven WORLD vocoder (FD-WORLD). It constructs frame-level spectra using spectral envelope priors, harmonic excitation and noise excitation, and generates waveforms via short-window inverse Fourier transform and overlap-add, boosting the audio synthesis speed of the vocoder by about 5 times. In experiments, Atom-TTS achieves an RTF of 0.0065 on a single-core CPU, corresponding to 155 times faster than real time. In a subjective listening test with 15 listeners on a shared corpus, it reaches a MOS of 3.24, outperforming a same-corpus baseline with 3.2 times more parameters by 1.32 MOS and another with 9.7 times more parameters by 0.99 MOS. The backend replacement is not distinguishable under natural listening. 80.5% of trials are judged as having no audible difference. Atom-TTS thus achieves a balance among quality, model size and efficiency.

## 2 INTRODUCTION

Offline navigation, device voice prompts, and voice assistance in network-constrained environments require text-to-speech systems to run under limited hardware resources, sharing CPU and memory with the primary application while each voice needs its own model and linguistic resources. Lightweight design must therefore cover the whole pipeline, yet existing methods struggle to model high-quality intermediate representations with few parameters, and well-designed vocoders still add substantial inference overhead on edge devices.

Neural speech synthesis has reached a high level of naturalness, and mainstream approaches generally adopt a two stage architecture: the front end predicts an intermediate representation and the back end converts it into a waveform (Wang et al., 2017; Ren et al., 2021; Kim et al., 2021; Mehta et al., 2024). Improvements in quality have largely come from scaling up both stages, so that larger acoustic networks and more expensive waveform generation modules are accompanied by higher inference cost (van den Oord et al., 2016; Kalchbrenner et al., 2018; Kong et al., 2020). Existing lightweight efforts have mostly optimized one component at a time (Luo et al., 2021; Valin & Skoglund, 2019), but the two are not independent: the output of the front end determines what the back end must process, and the choice of the back end in turn constrains the form of the representation the front end has to produce. Compressing either component in isolation therefore leaves the other as the bottleneck. This work instead fixes the intermediate representation first and then designs the two stages around it.

Mel spectrograms are the most widely used intermediate representation, typically consisting of 80 dimensional frame-level features (Ren et al., 2021). They discard phase and excitation information, so a separate neural vocoder must recover these from the data. This division of labour works when



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

resources are sufficient, but it raises the cost of both components on edge devices: the front end must predict a complete dense spectrogram, which limits how far the network capacity can be reduced, and the back end must carry a vocoder whose parameter count often exceeds that of the acoustic model while progressively upsampling back to the waveform, making it typically the most expensive part of the pipeline.

Parametric speech representations provide an alternative. Rather than using a dense spectrogram to fully describe each frame, they characterize speech with a small number of variables that have clear acoustic meanings. Among such representations, the decomposition into fundamental frequency, spectral envelope, and aperiodicity has been widely adopted. Statistical parametric speech synthesis methods employed this decomposition at an early stage (Zen et al., 2009), while the WORLD analysis and synthesis system provides a mature implementation of this approach (Morise et al., 2016). The spectral envelope captures timbre and resonance structure, the fundamental frequency describes periodic variations in voiced segments, and the aperiodicity parameters characterize noise components. Compared with mel spectrograms, these parameters have much lower dimensionality and inherently encode excitation and periodicity information. Part of the waveform can therefore be reconstructed through a structured synthesis process, allowing the neural network to focus on parameter prediction without reconstructing fine spectral details.

This work adopts the WORLD decomposition primarily for two advantages in lightweight speech synthesis. First, the front end only needs to learn a compact mapping from linguistic inputs to acoustic parameters, without requiring the network to simultaneously model detailed waveform information. Second, the back end can directly reconstruct the waveform from these parameters, without relying on latent variables or an additional neural vocoder. However, the decomposition itself does not address the computational cost of waveform generation. WORLD was originally designed for speech analysis and synthesis with high quality reconstruction. Its synthesis is organized around fundamental frequency pulses rather than fixed frames, so the computation grows with the pitch of the speech. On a single core CPU this becomes the bottleneck of the whole pipeline. Therefore, rather than using WORLD merely as a representation, this work retains its parameter interface while redesigning the frame-level synthesis process, so that the computational costs of both the front end and the back end remain within an acceptable range for edge devices.

This work proposes Atom-TTS, which addresses the two aforementioned challenges by reducing the parameter scale of the front end and the computational cost of waveform generation at the back end, respectively. First, we propose a Cascaded Parameter Acoustic Predictor, CPAP. It uses a TCN encoder to perform cascaded encoding of acoustic parameters and establishes explicit conditional dependencies among different parameters. CPAP reduces the number of trainable parameters to 161K without a noticeable degradation in prediction quality. We then propose a Frame-Driven WORLD vocoder, FD-WORLD, which constructs the spectrum of each frame using the spectral envelope prior together with harmonic and noise excitation, and then reconstructs the waveform through short window inverse Fourier transforms and overlap add operations.

The main contributions of this work are summarized as follows.

- We propose Atom-TTS, an ultra-lightweight TTS model that optimizes both acoustic parameter prediction and waveform synthesis, substantially improving synthesis efficiency while maintaining high quality audio generation.
- We propose CPAP, a cascaded parameter acoustic predictor that exploits explicit dependencies among acoustic parameters to achieve compact and efficient acoustic modeling with minimal quality degradation.
- We propose FD-WORLD, a frame-driven vocoder that reformulates WORLD synthesis at the frame level, substantially reducing waveform generation overhead while preserving the original parametric interface.
- We conduct extensive experiments to evaluate the proposed approach in terms of synthesis quality, model efficiency, vocoder efficiency, and edge device deployment, demonstrating its effectiveness for lightweight and real time speech synthesis.

# 3 Related Works

## 3.1 Parametric speech synthesis

Statistical parametric speech synthesis uses explicit acoustic variables to connect linguistic modeling with waveform generation (Zen et al., 2009). Generating smooth parameter trajectories requires continuity across frames, for which maximum likelihood parameter generation is a common approach (Tokuda et al., 1995), and spectral envelopes are often represented by mel generalized cepstral coefficients, which unify cepstral analysis and linear prediction (Tokuda et al., 1994). WORLD provides an analysis and synthesis framework built on fundamental frequency, spectral envelope, and aperiodicity, with CheapTrick, D4C, and Harvest supplying the corresponding analysis methods (Morise et al., 2016; Morise, 2015; 2016; 2017). Two departures from this formulation matter here. First, we represent the spectral envelope with mel generalized cepstral coefficients rather than WORLD's spectrogram-based envelope. Second, WORLD synthesis is driven by fundamental frequency pulses rather than fixed frames, so its cost grows with pitch and becomes a bottleneck on a single-core CPU. This motivates the frame-level redesign of the back end in Section 4.3. Our parallel-prediction baseline follows a Merlin-style system (Wu et al., 2016), whose parameter-matched comparison with the cascade is given in Section 5.3.

## 3.2 Lightweight acoustic modeling

Mapping phoneme sequences to frame-level representations typically requires explicit duration prediction, as in FastSpeech, while FastSpeech 2 further incorporates pitch and other variance information (Ren et al., 2019; 2021). To cut the cost of acoustic modeling, LightSpeech uses architecture search to obtain a smaller acoustic network (Luo et al., 2021), and temporal convolutional networks, gated convolutions, and depthwise separable convolutions provide parameter-efficient building blocks (Bai et al., 2018; Dauphin et al., 2017; Chollet, 2017). These methods mainly reduce network complexity through architectural design while making limited use of the inherent dependencies among parametric representations. Atom-TTS combines duration expansion with lightweight temporal convolutions, predicting acoustic parameters progressively according to their dependencies, and uses controlled comparisons under the same parameter budget to examine the effect of this organization.

## 3.3 High-efficiency waveform generation

Waveform generation mainly follows two directions. One uses neural networks directly or in combination with signal processing: WaveNet and WaveRNN generate waveforms autoregressively sample by sample, achieving high quality at the cost of parallelism and inference speed (van den Oord et al., 2016; Kalchbrenner et al., 2018), which motivated parallel generation such as HiFi-GAN's adversarial training (Kong et al., 2020), while LPCNet combines linear prediction with neural networks and focuses neural modeling on the residual signal (Valin & Skoglund, 2019). The other direction reconstructs the waveform in the frequency domain, as in iSTFTNet, which incorporates the inverse short-time Fourier transform into the vocoder, and Vocos, which predicts Fourier coefficients directly (Kaneko et al., 2022; Siuzdak, 2024). Both directions still rely on neural networks to obtain spectral or waveform representations, and their model size and inference cost tend to grow with quality requirements. Atom-TTS is related to the latter direction, but rather than predicting the spectrum with a neural network it constructs each frame spectrum directly from the predicted spectral envelope, fundamental frequency, and aperiodicity, then reconstructs the waveform through frame-level inverse transformation and overlap-add, keeping the back-end cost low.

# 4 Approach

## 4.1 Overall Structure and Acoustic Parameter Interface

Atom-TTS consists of a Cascaded Parameter Acoustic Predictor (CPAP) and a Frame-Driven WORLD vocoder (FD-WORLD). These two components interface via a shared set of acoustic parameters to form a complete text-to-waveform pipeline as Figure 1 shows. Text is converted by a linguistic front-end into a sequence of phonemes and their intra-phrase positions. Based on this, the

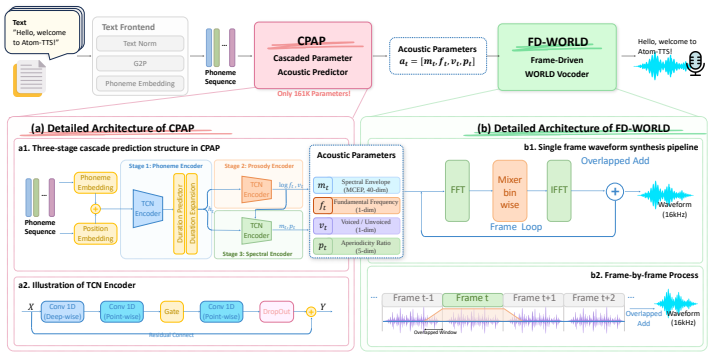


Figure 1: Overall architecture. Text becomes a phoneme sequence in the front end, CPAP predicts frame-level acoustic parameters, and FD-WORLD reconstructs the waveform. **(a)** Detailed architectural of CPAP. **(b)** FD-WORLD's frame-driven synthesis pipeline.

CPAP predicts frame-level acoustic parameters, and the FD-WORLD synthesizes these parameters into a waveform. The acoustic parameters for frame $t$ are denoted as

$$\mathbf{a}_t = [\mathbf{m}_t,\ f_t,\ v_t,\ \mathbf{p}_t], \qquad t = 1, \ldots, T, \tag{1}$$

Here, $T$ denotes the number of frames, and $\mathbf{m}t \in \mathbb{R}^{40}$ denotes the MCEP coefficients obtained through mel generalized cepstral analysis (Tokuda et al., 1994), which are used to reconstruct the spectral envelope of each frame. $f_t \in \mathbb{R}\geq 0$ denotes the fundamental frequency in Hz, with unvoiced frames assigned a value of 0. $v_t \in 0, 1$ denotes the voicing flag. $\mathbf{p}_t \in [0, 1]^5$ denotes the aperiodicity ratios of five frequency bands. Throughout the following sections, a hat denotes a predicted value, such as $\hat{\mathbf{m}}_t$, while symbols without a hat denote the corresponding ground truth values obtained from speech analysis.

For back end synthesis, let $E_t[k]$ denote the spectral envelope of frame $t$, where $k = 0, \ldots, 256$ is the frequency bin index. The spectral envelope is first decoded on a 2048 point grid to obtain sufficient frequency resolution, and is then sampled onto a 512 point grid according to $E_t[k] = E_t^{2048}[4k]$. Throughout this paper, $k$ consistently denotes the frequency bin index on the 512 point grid.

These parameters correspond directly to the synthesis process. $\mathbf{m}_t$ determines the spectral magnitude, $f_t$ and $v_t$ determine the locations and presence of harmonics, and $\mathbf{p}_t$ determines the ratio between harmonic and noise components. Therefore, the prediction network only needs to output a small number of low dimensional parameters, while the synthesis process does not need to infer the excitation structure from the data. This allows the two components to be independently compressed without imposing mutual constraints.

### 4.2 Cascaded Parameter Acoustic Predictor

CPAP performs calculations sequentially across three stages (phoneme structure, frame-level prosody, and spectral parameters), implemented by the Phoneme Encoder, the Prosody Encoder, and the Spectral Encoder, respectively, rather than using a single decoder to output all parameters simultaneously. This organization is motivated by the interdependency between spectral parameters and prosody. Unvoiced segments should lack harmonic structure, whereas the harmonic spacing in voiced segments is determined by the fundamental frequency.Consequently, the prediction of $\mathbf{m}_t$ and $\mathbf{p}_t$ must be conditioned on the frame's $f_t$ and $v_t$. By first determining the temporal structure, then the prosody, and finally predicting the spectral parameters based on the known prosody, the spectral parameter stage avoids the need to re-infer the temporal structure.

**Three stages.** In the phoneme stage, phoneme IDs concatenated with embeddings of their relative positions within the phrase are fed into the Phoneme Encoder, a temporal convolutional network, to obtain phoneme-level representations $\mathbf{h}_{1:P}$ (where $P$ is the number of phonemes). A duration prediction head outputs the number of frames $\hat{d}_P$ for each phoneme, using forced alignment re-

sults (McAuliffe et al., 2017) as supervision during training and the predicted values $\hat{d}_{1:P}$ during inference. In the frame-level stage, $\mathbf{h}_{1:P}$ is expanded into a frame-level sequence based on $\hat{d}_{1:P}$, while the intra-phoneme relative position $r_t = t - s(t)$ is generated for the $t$-th frame, with $s(t)$ denoting the starting frame index of the phoneme to which the frame belongs. $r_t$ is truncated at 255 and embedded as $\mathbf{q}_t \in \mathbb{R}^8$ to capture intra-phoneme dynamic changes, such as the gradual spectral transition from a consonant to a vowel. The expanded sequence is concatenated with $\mathbf{q}_t$ and fed into the Prosody Encoder to obtain $\mathbf{x}_{1:T}$, followed by a fundamental frequency head and a voiced/unvoiced head that output $\hat{f}_t$ and $\hat{v}_t$. In the spectral parameter stage, $[\hat{f}_t, \hat{v}_t]$ is mapped to a condition vector $\mathbf{c}_t \in \mathbb{R}^{24}$, which is concatenated with $\mathbf{x}_{1:T}$ and fed into the Spectral Encoder to output $\hat{\mathbf{m}}_{1:T}$ and $\hat{\mathbf{p}}_{1:T}$.

**Basic computational unit.** Each of the three encoders is composed of gated depthwise separable temporal convolutions (Bai et al., 2018; Dauphin et al., 2017; Chollet, 2017). Within each block, a depthwise convolution aggregates temporal context and is applied with dilation, the dilation rate doubling at every layer so that the receptive field grows exponentially with depth without adding parameters. Pointwise convolutions then mix channels, with the first expanding them twofold so that a gated linear unit can split the result into a value and a gate. The gate multiplies the value and the second pointwise convolution projects back to the original width. This split gives depthwise convolutions the small parameter budget ($k \times C$ per layer) that makes the block cheap, while the gating restores the expressiveness that pure depthwise filtering would lose. The design keeps a narrow network (96 channels) within a sufficient contextual span at a low parameter count, and it supports the capacity experiment design in Section 5.3, which varies depth at constant width. Prosody post-processing at inference (dual-threshold voicing decision, moving average, and pause masking) is described in Appendix 11.1.4.

**Parameter count and depth configuration.** CPAP has 161K trainable parameters, of which the Prosody Encoder holds about 43% because it carries both frame-level encoding and the heads for fundamental frequency and voicing, against roughly 29% for the Phoneme Encoder and 28% for the Spectral Encoder. Encoder depth is identical across the three stages and denoted $d$. The main configuration is $d = 1$, and the capacity experiment raises it to $d = 3, 5, 7, 9$, giving 335K, 508K, 681K, and 855K parameters. Quality gains are reported in Section 5.3 and the full breakdown in Appendix 11.1.1.

### 4.3 Frame-Driven WORLD vocoder

FD-WORLD retains the acoustic parameter interface of WORLD but replaces pulse-driven synthesis, which iterates over fundamental frequency pulses, with synthesis driven by a fixed frame rate. This is the primary source of the back-end acceleration: since $\mathbf{m}_t$, $f_t$, $v_t$, and $\mathbf{p}_t$ already specify the spectral envelope, harmonic positions, and noise ratio, constructing the spectrum frame by frame removes the need for temporal interpolation or phase estimation.

**Motivation.** WORLD iterates over all F0 pulses in its outer loop. For each pulse it interpolates the spectral envelope and aperiodicity at the exact pulse instant and constructs the periodic component under a minimum-phase assumption. Since the pulse count grows with F0, synthesis cost rises markedly for high-pitched speech. FD-WORLD instead drives the outer loop by frame, so the iteration count equals the number of frames and is independent of F0. Parameters within a frame come from that frame's predictions, requiring no fractional-position interpolation. And the periodic excitation is written directly as a pulse train in the frequency domain, with phase determined naturally by pulse positions. The trade-off is lower phase precision than the pulse-by-pulse approach, but at 200 frames per second the frame rate exceeds typical F0 values, so the perceptual difference is negligible (Section 5.4).

**Synthesis rules.** Let $Y_t[k]$ denote the synthesized spectrum of frame $t$. With $k$ and $E_t[k]$ as defined in Section 4.1,

$$Y_t[k] = \sqrt{E_t[k]}\Big((1 - a_t[k])\, P_t[k] + a_t[k]\, \widetilde{N}_t[k]\Big), \tag{2}$$

Here $a_t[k] \in [0, 1]$ is the per-bin noise mixing weight, obtained by expanding the five sub-band values of $\mathbf{p}_t$ onto the 257-bin grid of the synthesis transform (all bins inside a sub-band take that band's value). $P_t[k]$ is the spectrum of the periodic excitation: the pulse train controlled by $f_t$ and

Table 1: Subjective quality on the same corpus; labels were concealed and samples balanced, with 15 listeners each rating 20 sentences per system across four systems (1,200 MOS ratings), and values are means with 95% confidence intervals from a paired test with intra-listener averaging. Listening was conducted as the naturalness stage of a three-stage session that also contained an ABX and a switch-detection task.

| System | Back-end vocoder | Params | MOS ↑ | 95% CI |
|---|---|---|---|---|
| Piper alan-medium (reference) | VITS | 15,650,556 | 3.87 | [3.46, 4.29] |
| sanoTTS standard | iSTFT | 1,567,244 | 2.25 | [2.00, 2.50] |
| sanoTTS tiny | iSTFT | 512,571 | 1.92 | [1.65, 2.18] |
| **Atom-TTS (ours)** | **FD-WORLD** | **161,288** | **3.24** | **[2.87, 3.61]** |

Table 2: Deployment cost and intelligibility. WER is a cross-system reference only, since the external systems use different corpora, while RTF and net memory use a unified protocol: single-threaded onnxruntime CPU, inference only, excluding loading and warm-up.

| System (back-end vocoder) | Params | WER ↓ | RTF ↓ | Net memory (MB) |
|---|---|---|---|---|
| Kokoro-82M (ISTFTNet) (Hexgrad, 2024) | 81,763k | 0.0067 | 0.57984 | 338.7 |
| Piper alan (VITS) (Rhasspy, 2023) | 15,650k | 0.0052 | 0.19565 | 82.3 |
| KittenTTS nano (iSTFTNet) (KittenML, 2025) | 14,009k | 0.0267 | 0.27449 | 73.8 |
| TinyTTS (VITS) (TinyTTS Contributors, 2025) | 1,618k | 0.0097 | 0.02589 | 33.4 |
| sanoTTS standard (iSTFT) (Thapa, 2026) | 1,567k | 0.0166 | 0.03300 | 36.3 |
| EfficientSpeech (HiFi-GAN) (Atienza, 2023) | 1,192k | 0.0128 | 0.23144 | 126.5 |
| sanoTTS tiny (iSTFT) (Thapa, 2026) | 512k | 0.0136 | 0.01994 | 27.3 |
| **Atom-TTS (FD-WORLD, ours)** | **161k** | **0.0173** | **0.00645** | **7.0** |

$v_t$ is windowed and transformed frame by frame. And $\widetilde{N}_t[k]$ is the amplitude-scaled spectrum of the noise track. The factor $\sqrt{E_t[k]}$ matches the amplitude scaling used inside WORLD, since a linear envelope would systematically suppress high-frequency energy. When $v_t = 0$ the pulse train is empty and the frame is entirely noise, so unvoiced frames need no separate branch.

For a single sentence, the spectral envelope is decoded from $\mathbf{m}_{1:T}$ and resampled onto a 512-point grid by keeping every fourth bin of the 2048-point WORLD decode. Noise weights are expanded from $\mathbf{p}_{1:T}$ across sub-bands. A pulse train is generated by phase accumulation from $f_{1:T}$. A noise track is generated from a fixed seed. Each frame is then computed by Eq. 2 and inverse-transformed, and the waveform is obtained by overlap-add with a Hann window and normalization by the summed squared window. The full procedure is given in Appendix 11.1.5.

**Implementation details.** The inverse transform uses a 512-point grid with a frame shift of 120 samples, giving about 77% overlap between adjacent frames. The short analysis window makes the duration covered by one frame (about 21 ms) exceed the frame shift, which smooths the envelope after overlap-add and keeps the spectral envelope changing slowly enough that intra-frame parameters can be treated as constant. The back end has no trainable parameters, and its cost scales with the number of frames rather than with fundamental frequency.

### 4.4 Training Objective

The individual components of the CPAP are directly predicted by corresponding prediction heads, using ground-truth parameters extracted from the speech signal as supervision during training:

$$\mathcal{L} = \lambda_d \mathcal{L}_d + \lambda_f \mathcal{L}_f + \lambda_v \mathcal{L}_v + \lambda_m \mathcal{L}_m + \lambda_p \mathcal{L}_p, \tag{3}$$

Here, $\mathcal{L}_d$ is the mean squared error of log-duration, $\mathcal{L}_f$ is the fundamental frequency error, $\mathcal{L}_v$ is the binary cross-entropy for the voiced/unvoiced decision, and $\mathcal{L}_m$ and $\mathcal{L}_p$ correspond to MCEP and aperiodicity parameters, respectively. In addition to frame-wise error, the MCEP loss constrains the first- and second-order differences and includes a variance-preservation term to prevent over-

smoothing of the output, which would otherwise result in the loss of formant details. The specific weight values are provided in Appendix 11.1.3.

The spectral parameter stage relies on the predicted $\hat{f}_t$ and $\hat{v}_t$. However, since these predictions contain significant errors during the early stages of training, using them directly would interfere with the learning of the Spectral Encoder. To address this, a scheduling strategy is employed during training: ground-truth prosody is selected with probability $p_n$, while predicted prosody is selected with probability $1 - p_n$, with $p_n$ linearly decaying from 1.0 to 0.2 over the first 120 epochs. This approach ensures that the decoder encounters an input distribution consistent with inference during the later stages of training, without requiring reliance on ground-truth prosody at inference time.

# 5 Experiments

## 5.1 Data, Reference Systems, and Evaluation Protocol

**Data and systems.** Experiments use an English single-speaker corpus (Piper's neutral British male voice, alan-medium): 7,974 training and 1,993 test utterances at 24 kHz, evaluated on 50 sentences. Reference systems split into those trained or distilled on the same corpus, whose quality is directly comparable, and public systems that use different corpora and participate only in the comparison of parameters, speed, and memory. Details are in Appendix 11.2.

**Evaluation.** Naturalness uses a five-point MOS following the standard for subjective speech evaluation (ITU-T, 1996), over the four systems trained on the same corpus: 15 listeners each rated the same 20 test sentences of every system, giving 1,200 MOS ratings, with labels concealed and samples balanced, and Piper alan-medium serves as the strongest same-corpus baseline since the corpus has no human recordings. Comparisons use paired tests with 95% confidence intervals and a non-inferiority margin of $\delta = 0.3$ MOS frozen before listening. Intelligibility uses WER from faster-whisper large-v3 (Radford et al., 2023), and prediction quality uses MCD, F0 RMSE, voiced/unvoiced error rate, and duration MAE. Internal capacity comparisons additionally report UTMOS (Saeki et al., 2022) and DNS-OVRL (Reddy et al., 2022), which are used only to compare configurations under the same back end and acoustic objective, never to rank systems across corpora.

**Cost and statistics.** Inference speed and memory follow a unified protocol (single-threaded on an Intel i7-12700, 16GB RAM, win10 system with onnxruntime, inference only, memory as the resident set against an empty baseline), with further measurement details in Appendix 11.2.2. Objective comparisons use per-sentence paired bootstrap over the same test sentences, reporting differences, 95% confidence intervals, $p$ values, and effect sizes $d_z$. A non-significant result alone is not treated as equivalence.

## 5.2 Quality and Cost of the Complete System

**Subjective quality.** Table 1 shows that this work exceeds sanoTTS tiny by 1.32 MOS using 31% of its parameters ($p = 6.9 \times 10^{-6}$, $d_z = 1.79$) and sanoTTS standard, which has 9.7 times more parameters, by 0.99 MOS using 10% of its parameters ($p = 1.9 \times 10^{-5}$). All 15 listeners rate it higher in both comparisons. Against Piper alan it reaches 84% of the reference's naturalness with 1.03% of its parameters, a gap of 0.63 MOS. The confidence intervals of the two baselines do not overlap with ours at all, so the differences are not attributable to listener variation, whereas the partial overlap with the reference reflects a limited gap. Inter-listener agreement is ICC(2,1) = 0.396: listeners differ in absolute scale but agree closely on ranking.

**Deployment cost.** In Table 2, this work has the smallest parameter count (161K, 3.2 times smaller than the next smallest and 507 times smaller than Kokoro-82M) and the lowest RTF and memory: an RTF of 0.00645, 155 times faster than real time and 3.1 times faster than the next fastest system, with a net memory increase of 7.0 MB against 27.3 to 338.7 MB elsewhere. Because our text front end runs inside the C++ runtime and does not occupy Python process memory, the memory comparison is based on the net increase, which isolates the cost attributable to each system.

**Intelligibility.** Table 2 adds a WER column. Atom-TTS obtains 0.0173, which is higher than several systems in the table but the same order of magnitude as the reference we compare against most directly. Due to factors such as timbre and speech data, the value of 0.0173 is therefore best read as

Table 3: Parameter-matched comparison of cascade and parallel prediction (161,208 against 160,564 parameters, 0.4% apart, with identical data, objective, and back end); paired tests over 300 utterances give $p < 10^{-19}$, and the sign of $d_z$ is unified so that positive favours the cascade.

| Metric | Cascade (ours) | Parallel prediction | Δ | $d_z$ |
|---|---|---|---|---|
| MCD full band (dB) ↓ | **5.863** | 6.169 | +0.306 | 2.11 |
| MCD low band ↓ | **4.859** | 5.155 | +0.296 | 1.98 |
| MCD high band ↓ | **1.925** | 1.973 | +0.048 | 2.52 |
| Envelope preservation, mid ↑ | **0.756** | 0.713 | −0.043 | **−5.12** |
| Envelope preservation, high ↑ | **0.595** | 0.544 | −0.051 | **−7.62** |
| F0 RMSE (cents) ↓ | 119.7 | **116.8** | −2.88 | −0.53 |

evidence that the system incurs no severe intelligibility loss, rather than as a quality score comparable to values such as 0.0052.

### 5.3 Capacity and Architecture Ablation

All configurations share the same training pipeline and FD-WORLD back end, so the two comparisons below isolate front-end design choices.

**Parameter budget.** Appendix A3 reports the full capacity ablation. Increasing the parameter count by 2.07 times ($d1$ to $d3$) yields only a 0.29 gain in UTMOS, after which returns largely stop: a further 2.56-fold increase to $d9$ changes UTMOS by at most 0.05. Compute grows from 0.114 M multiply-adds per frame at $d1$ to 0.590 M at $d9$, yet end-to-end RTF rises by only 38%, so total time in this range is dominated by back-end synthesis and front-end capacity is not the bottleneck. Figure 2(a) confirms the plateau: the validation MCD of the five settings draws together after $d3$ and the gap does not change as training continues.

**Cascade versus parallel.** In Table 3, parallel prediction shares one decoder and trains faster, but the cascade lowers full-band MCD by 0.306 dB consistently across the low, mid, and high bands. The effect sizes show where the gain lives: envelope preservation reaches −5.12 and −7.62 in the mid and high bands, far exceeding the 2.11 for full-band MCD, so the improvement is concentrated in spectral envelope fidelity rather than a uniform reduction of all errors. Parallel prediction holds only a slight edge in fundamental frequency (2.88 cents lower RMSE). Figure 2(a) shows the parallel MCD sitting alone above the rest with a stable gap from epoch 400 onward, so the difference stems from the architecture rather than from insufficient training.

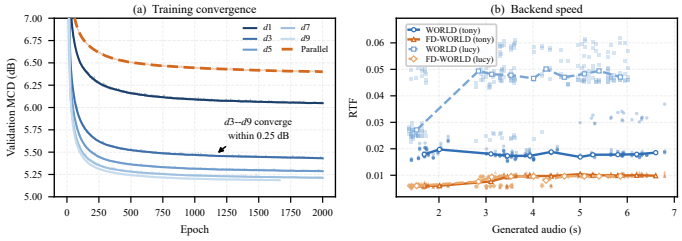


Figure 2: Ablation results. **(a)** Validation MCD for $d1$, $d3$, $d5$, $d7$, $d9$, and parallel prediction, where thin lines are per-epoch values, thick lines moving averages, and the dashed line parallel prediction. **(b)** Back-end RTF against audio duration for {WORLD, FD-WORLD} × {male, female}. WORLD shifts with voice while FD-WORLD does not.

### 5.4 Back-End Comparison

Both back ends receive exactly the same CPAP output, so any difference comes entirely from the process of reconstructing a waveform from the parameters.

**Speed.** In Table 4, given the same front-end output, FD-WORLD reduces vocoder time from 164.01 ms to 25.73 ms and end-to-end RTF from 0.04713 to 0.00950. The time breakdown shows that the cost of pulse-driven synthesis is almost entirely concentrated in the vocoder.

Table 4: Back-end comparison; both settings take the same front-end output and differ only in the back end, which has no trainable parameters in FD-WORLD. ABX no-difference is the share of trials judged indistinguishable, and the masking rate is the share of difference energy below the threshold.

| Back end | Vocoder time | End-to-end RTF | ABX no-difference | Masking rate | WER |
|---|---|---|---|---|---|
| WORLD | 79.39 ms | 0.03446 | — | — | 0.0135 |
| **FD-WORLD** | **16.44 ms** | **0.00645** | **80.5%** | **99.1%** | **0.0173** |

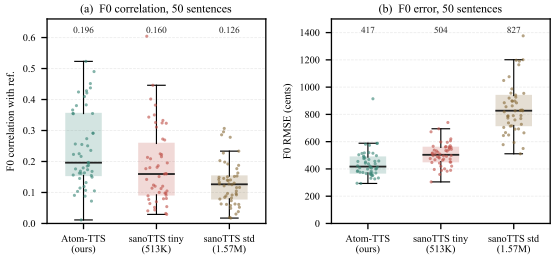


Figure 3: Fundamental frequency analysis over 50 sentences. **(a)** F0 correlation against the reference, with medians annotated. **(b)** F0 root mean square error (cents); F0 is extracted by the same estimator for all systems.

Figure 2(b) shows the per-sentence RTF distribution for the male (tony) and female (lucy) voices. The cost of WORLD is proportional to the number of pitch pulses, so the female voice (higher F0, denser pulses) needs markedly more vocoder time (164.01 ms) than the male voice (63.48 ms). FD-WORLD is driven by a fixed frame rate, so the two voices take similar time (25.73 and 30.96 ms) and the speedup is 6.37 times for the female voice against 2.05 times for the male. The benefit of FD-WORLD is therefore larger for high-pitched voices, while for low-pitched voices the absolute time remains below 32 ms.

WORLD estimates a minimum-phase response for each pitch pulse and interpolates the envelope and aperiodicity at the exact pulse instant, whereas FD-WORLD uses the natural phase of the pulse train and holds parameters constant within a frame, which removes per-pulse phase estimation and time interpolation at the cost of no longer controlling phase explicitly.

**Perceptual equivalence.** In the within-sentence switching task, Table 4 shows that 80.5% of trials were judged indistinguishable and forced-choice accuracy does not differ significantly from chance, with 99.1% of the difference energy falling below the masking threshold. A detection task aimed specifically at the switch point still finds a detection rate 12.7 percentage points above the negative control, so the difference does not vanish entirely, though its level is below one quarter. The two back ends are therefore equivalent under ordinary listening.

**Prosody consistency.** Over the 50 test sentences, the median F0 correlation of this work is 0.196, above 0.160 for sanoTTS tiny and 0.126 for sanoTTS std, and the root mean square error is 417 cents against 504 and 827 cents (Figure 3). All F0 values come from the same estimator.

## 6 Conclusion

This work presents Atom-TTS, a lightweight speech synthesis system for a single-core CPU. Its front end, the Cascaded Parameter Acoustic Predictor, predicts the full set of WORLD acoustic parameters with only 161K trainable parameters, and its back end, the frame-driven WORLD vocoder, reconstructs the waveform deterministically with no trainable parameters at all. Because both sides share one parametric interface, they can be compressed separately rather than shifting cost from one to the other. On a single-core CPU the system reaches an end-to-end RTF of 0.00645, about 155 times faster than real time, with a net memory increase of 7.0 MB, and it obtains a MOS of 3.24 that exceeds same-corpus baselines several times its size. These results show that a parametric interface with a deterministic back end can deliver competitive naturalness at a fraction of the usual cost.

## 7 Limitations

The current system is trained per voice and relies on a text front end and phoneme-level duration supervision, so the representational capacity of the acoustic parameters and the prediction error jointly determine final quality. Cross-voice style control, richer prosodic expression, and validation on more devices are the natural extensions.

## 8 AI Usage Statement

Generative artificial intelligence was used in preparing this manuscript to help organize existing material, structure the text.

## 9 Ethics Statement

**Speech data and licensing.** The training and evaluation corpus used in this work is not a recording of human speakers: it was generated with the open-source Piper text-to-speech system (alan-medium voice), distributed under the MIT licence, and the underlying voice model is a neutral British male voice whose release permits synthesis of arbitrary text. Because the corpus consists of synthesized speech, no recordings of identifiable individuals were collected, and no speaker consent was required for its use. Systems trained on other corpora (Kokoro-82M, KittenTTS, TinyTTS, EfficientSpeech, and the sanoTTS variants) are used under their respective open licences, and are evaluated only for engineering quantities, parameter count, inference time, and memory, because their training data differ from ours and quality comparisons across corpora would not be meaningful.

**Speaker rights.** No voice cloning, speaker adaptation, or identity transfer is performed. The system is trained per voice from a single licensed voice model, and it neither extracts nor reproduces the identity of any real speaker beyond the timbre already present in that model. We do not release any synthetic speech that impersonates a specific named individual.

**Listening tests and participant privacy.** The subjective evaluation was conducted with 15 volunteer listeners. Each listener completed 184 trials in a three-stage session: 84 naturalness ratings (20 sentences $\times$ 4 systems, plus 4 attention checks), 60 ABX trials, and 40 switch-detection trials, giving 2,760 responses in total. Participation was voluntary and unpaid, no personal or demographic information was collected, and no audio was recorded from participants. Responses were stored without identifiers, and system labels were concealed and sample order balanced so that listeners could not infer which system they were rating. The protocol follows the ITU-T P.800 recommendation for subjective quality assessment. Since the study collected no personally identifiable information and involved no intervention on participants, it fell outside the scope of formal ethics review at our institution.

## 10 Reproducibility Statement

The appendix collects the material needed to reproduce every reported number, and the references below point to it. Data sizes, splits, and the evaluation set are in Appendix 11.2.1. The parameter breakdown, the layer and dilation configuration of each capacity setting with its training cost, and the loss weights are in Appendices 11.1.1, 11.1.2, and 11.1.3. The frame-level synthesis procedure is given as pseudocode in Appendix 11.1.5, preceded by the prosody post-processing in Appendix 11.1.4; the back end is deterministic and can therefore be reimplemented from these two sections alone. The measurement protocol for speed and memory is in Appendix 11.2.2, the reference systems in Appendix 11.2.3, and the automatic metrics in Appendix 11.2.4, while the subjective protocol is described in Section 5.1. All diagnostic and consistency data behind the back-end claims are in Appendices 11.3.1–11.4.

Table A1: Parameter breakdown of CPAP (main configuration $d = 1$, 102 English phonemes)

| Stage | Main components | Parameters | Share |
|---|---|---|---|
| Phoneme Encoder | 47,393 | 29.4% | |
| Prosody Encoder | 68,890 | 42.7% | |
| Spectral Encoder | 44,925 | 27.9% | |
| **Total** | **161,208** | 100% | |

Table A2: Layer counts and training cost of the capacity settings

| Configuration | Layers $d$ | Dilation sequence | Parameters | Epochs | Total time (h) |
|---|---|---|---|---|---|
| $d1$ | 1 | $[1]$ | 161,288 | 2000 | 13.81 |
| $d3$ | 3 | $[1, 2, 4]$ | 334,584 | 2000 | 12.35 |
| $d5$ | 5 | $[1, 2, 4, 8, 16]$ | 507,960 | 2000 | 14.75 |
| $d7$ | 7 | $[1, \ldots, 64]$ | 681,336 | 2000 | 16.37 |
| $d9$ | 9 | $[1, \ldots, 256]$ | 854,712 | 1664 | 10.73 |
| parallel | 1 | $[1]$ | 160,564 | 2000 | 7.42 |

# 11 Appendix: Supplementary Material on Implementation and Evaluation

## 11.1 A. Implementation Details

### 11.1.1 A.1 Parameter Breakdown of CPAP

Section 4.2 reports a total of 161K trainable parameters for CPAP. This section gives the exact breakdown by stage.

### 11.1.2 A.2 Depth Configuration and Training Cost

The training corpus is `en/male_tony`, with 7,974 utterances and about 17.09 million frames. Batch size, learning rate, and optimizer are identical across all configurations.

### 11.1.3 A.3 Training Loss Weights

Section 4.4 gives the form of the loss. The individual weights are set as follows: duration $\mathcal{L}_d$ 0.8, F0 $\mathcal{L}_f$ 1.0, voiced/unvoiced $\mathcal{L}_v$ 0.1, aperiodicity $\mathcal{L}_p$ 0.25, the zeroth-order MCEP coefficient 0.2, the MCEP spectral shape 1.75, the first and second MCEP differences 0.75 and 0.2, variance preservation 0.1, and the frequency-domain spectral envelope 0.05. The MCEP terms carry most of the weight because the spectral envelope dominates perceived quality, while the zeroth-order and variance terms are kept small so that they regularize without competing with the shape term.

### 11.1.4 A.4 Prosody Post-Processing

Deciding voiced/unvoiced probabilities by a single threshold makes the decision flip repeatedly near boundaries, which is heard as intermittent clicks. At inference the V/UV probabilities are first smoothed with a 5-frame moving average, after which a two-threshold decision is applied: the state turns voiced above 0.55 and unvoiced below 0.35, and in between it retains the previous state. Isolated unvoiced runs of at most 2 frames are then filled in. The fundamental frequency is smoothed in the same way and multiplied by the V/UV flag so that $f_t = 0$ in unvoiced frames. In addition, pause frames corresponding to punctuation are forced to unvoiced with aperiodicity set to 1.0, and the zeroth MCEP coefficient is clamped to at least $-35$ so that no audible vowel residue remains at pause positions.

Table A3: CPAP capacity ablation, all relative to $d1$, where each setting changes only the layer count $d$ of the three encoders; parameter efficiency is UTMOS per 100K parameters, normalized to 1.00 at $d1$, and UTMOS is used only for internal comparison under the same back end.

| Configuration | Params | Relative params | MACs/frame | ΔUTMOS | ΔRTF | Parameter efficiency |
|---|---|---|---|---|---|---|
| $d1$ (main) | 161,288 | 1.00× | **0.114 M** | — | — | **1.00** |
| $d3$ | 334,584 | 2.07× | 0.233 M | +0.29 | −3% | 0.51 |
| $d5$ | 507,960 | 3.15× | 0.352 M | +0.33 | +8% | 0.32 |
| $d7$ | 681,336 | 4.22× | 0.471 M | +0.29 | +29% | 0.23 |
| $d9$ | 854,712 | 5.30× | 0.590 M | +0.25 | +38% | 0.19 |

Table A4: Algorithm of frame-level parametric synthesis; $\mathbf{h}$ is a length-512 Hann window, $\odot$ and $\oslash$ elementwise multiplication and division, and $\epsilon$ prevents division by zero, while lines 5 and 10–13 implement windowed overlap-add with normalization by the summed squared window.

| Line | Operation |
|---|---|
| Input | $\mathbf{m}_{1:T}$, $f_{1:T}$, $\mathbf{p}_{1:T}$, random seed $s$ |
| Output | waveform $\mathbf{y}$ |
| 1 | $E_{1:T} \leftarrow$ decode spectral envelope from $\mathbf{m}_{1:T}$, resample to a 512-point grid by keeping every fourth bin |
| 2 | $a_{1:T} \leftarrow$ expand $\mathbf{p}_{1:T}$ by sub-band onto the same 512-point grid |
| 3 | $H_{1:T} \leftarrow$ accumulate phase from $f_{1:T}$, place pulses at phase wraps |
| 4 | $\widetilde{N}_{1:T} \leftarrow$ generate Gaussian noise from seed $s$ and calibrate amplitude |
| 5 | $P_{1:T}[k], \widetilde{N}_{1:T}[k] \leftarrow$ windowed DFT of $H_{1:T}$ and $\widetilde{N}_{1:T}$ on 512-point frames |
| 6 | $\mathbf{y} \leftarrow \mathbf{0}, \mathbf{w} \leftarrow \mathbf{0}$ |
| 7 | **for** $t = 1$ **to** $T$ **do** |
| 8 | $\quad Y_t[k] \leftarrow$ compute per frequency bin using Eq. 2 |
| 9 | $\quad \mathbf{y}_t \leftarrow$ inverse transform of $Y_t$ |
| 10 | $\quad \mathbf{y}[t] \leftarrow \mathbf{y}[t] + \mathbf{y}_t \odot \mathbf{h}$ |
| 11 | $\quad \mathbf{w}[t] \leftarrow \mathbf{w}[t] + \mathbf{h} \odot \mathbf{h}$ |
| 12 | **end for** |
| 13 | **return** $\mathbf{y} \oslash \max(\mathbf{w}, \epsilon)$ |

#### 11.1.5 A.5 Algorithm of Frame-Level Synthesis

#### 11.1.6 A.6 Single-Core Deployment

**Deployment form.** CPAP is split into three ONNX subgraphs, with the split points chosen where control flow is required: duration expansion involves rounding and length replication, and prosody smoothing needs a cross-frame hysteresis decision, neither of which maps well onto a static graph, so both are implemented in the C++ runtime. The runtime also handles parameter denormalization, pause masking, and FD-WORLD synthesis. Model weights, statistics, and linguistic resources are packed into a single voice file, and inference is entirely single-threaded.

**Time breakdown.** Under the $d1$ configuration, the mean synthesis time per sentence is 34.7 ms, of which FD-WORLD synthesis accounts for 31.35 ms, that is more than 90%. The three ONNX subgraphs total 1.84 ms, the C++ glue layer 0.49 ms, and the linguistic front end 0.016 ms, so the three front-end items together are below 8%. This breakdown explains the behaviour seen in Table 3: increasing CPAP capacity has limited effect on end-to-end RTF because total time is dominated by the back end. It also means that further speed improvements should focus on synthesis rather than on the prediction network.

**Memory and startup.** After loading and warm-up in a separate process, Atom-TTS has a resident set of 49.6 MB, a net increase of 7.0 MB over the empty baseline. The empty baseline needs only ONNX Runtime and the WORLD library, 42.6 MB in total. Our text front end runs inside C++ and needs no Python-side G2P or phonemization dependency, so the net increase is lower than for the systems in Table 2 that depend on a Python front end.

Table A5: Training corpus of each reference system, and whether its quality metrics are comparable to ours

| System | Parameters | Training corpus | Comparable quality metrics |
|---|---|---|---|
| Atom-TTS (ours) | 161k | this work | ✓ |
| sanoTTS distilled tiny | 512k | this work | ✓ |
| sanoTTS distilled standard | 1,567k | this work | ✓ |
| Piper alan-medium | 15,650k | distillation reference | ✓ |
| EfficientSpeech | 1,192k | LJSpeech | × |
| sanoTTS heart-nano | 294k | official | × |
| TinyTTS | 1,618k | official | × |
| KittenTTS nano | 14,009k | official | × |
| Kokoro-82M | 81,763k | official | × |

**Consistency verification.** To ensure that the deployed implementation matches the trained model, three numerical comparisons were performed: the maximum absolute error per graph between PyTorch and ONNX under fixed inputs does not exceed $8 \times 10^{-6}$. The C++ implementation agrees with the Python reference front end on duration, energy, and amplitude spectrum statistics (mean log-spectral difference below 0.3 dB). And the directory-asset mode and single-file voice mode produce bit-identical output. The FD-WORLD noise track is generated from a fixed seed, so the waveforms in these comparisons are reproducible. Detailed records are given in Appendix 11.4.

## 11.2 B. Evaluation Protocol

### 11.2.1 B.1 Data and Splits

The training set has 7,974 utterances and the test set 1,993, with a mean duration of 10.7 s and a total of about 23.7 h. Evaluation text consists of 50 English sentences covering both short and long inputs.

### 11.2.2 B.2 Cost Measurement Protocol

The runtime is onnxruntime CPU 1.23.2, single-threaded (`intra_op_num_threads` and `inter_op_num_threads` both set to 1). The C++ portion of Atom-TTS is compiled with the MSYS2 ucrt64 toolchain and `-O3`. Timing covers inference only, excluding loading and warm-up. Each system is measured over 45 groups (9 sentences $\times$ 5 rounds $\times$ 30 repetitions, taking the median). Memory is the resident set after loading and warm-up in a separate subprocess.

**Effect of the compilation environment.** FD-WORLD requires one square root operation per frame per frequency bin, so its cost is sensitive to the implementation of the math library. The relative speedup measured under Linux with glibc differs from the protocol used here, so the speed results in the main text are tied to a Windows and ucrt64 environment.

### 11.2.3 B.3 Reference Systems

Systems marked × are comparable to this work in parameter count, speed, and memory, but their quality metrics are affected by corpus and recording conditions and are not used to judge quality.

### 11.2.4 B.4 Automatic Quality Metrics Across Systems

Table 1 reports only human MOS, which for reasons of listening budget covers just the four systems trained on the same corpus. This section gives four families of automatic metrics (UTMOS (Saeki et al., 2022), WER (Radford et al., 2023), SCOREQ, DNS-OVRL (Reddy et al., 2022)) for all comparable systems, using the same test set as the listening test (the 50 sentences of en/male_tony). Note that these four metrics measure different dimensions: WER reflects only intelligibility, UTMOS and SCOREQ reflect naturalness, and DNS-OVRL reflects overall signal quality. Consequently their rankings disagree for lightweight systems (for example heart-nano has a good WER but a very low UTMOS, as shown below).

Table A6: Automatic quality metrics across all comparable systems, on the same 50-sentence test set as the listening test

| System | UTMOS ↑ | WER ↓ | SCOREQ ↑ | DNS-OVRL ↑ |
|---|---|---|---|---|
| **Atom-TTS (FD-WORLD)** | 2.880 | 0.0173 | 2.710 | 3.343 |
| sanoTTS distilled tiny (same corpus) | 3.000 | 0.0136 | 3.306 | 3.819 |
| sanoTTS distilled standard (same corpus) | 3.477 | 0.0166 | 3.774 | 3.839 |
| Piper alan-medium (reference) | 4.066 | 0.0052 | 4.348 | 3.768 |
| EfficientSpeech | 3.574 | 0.0128 | 3.835 | 3.149 |
| TinyTTS | 3.824 | 0.0097 | 4.080 | 3.852 |
| KittenTTS nano | 4.341 | 0.0267 | 4.609 | 3.836 |
| Kokoro-82M | 4.508 | 0.0067 | 4.885 | 3.978 |

Table A7: Waveform-level differences between the two back ends

| Test sentence | SNR (dB) | Spectral distance (dB) | energy envelope correlation | Masked fraction |
|---|---|---|---|---|
| 01 | −1.14 | 7.6 | 0.956 | 99.13% |
| 04 | −2.84 | 6.4 | 0.969 | 99.15% |
| 44 | +0.08 | 21.3 | 0.966 | 64.01% |

The SCOREQ and DNS-OVRL values of the distilled sanoTTS variants were measured under the same protocol as the main table (same 50-sentence test set, same script). These two metrics do not agree with the ranking of human MOS. The reason is discussed in Appendix 11.3.2.

**WER measures transcription, not quality.** The WER column should be read with two caveats. First, the external systems were trained on different corpora, so their WER reflects both the model and the recording conditions of their training data. Second, WER only measures how easily an ASR model transcribes the output; it says nothing about naturalness, and several errors shared across systems are in fact ASR normalization differences rather than genuine recognition failures (for example travelled against traveled, or "over 200"). WER should therefore be used only to confirm that a system has no severe intelligibility loss, and not read as a quality score.

### 11.3 C. Analysis of Back-End Differences

Section 5.4 reported the speed gain and subjective quality of FD-WORLD relative to WORLD and noted that their differences concentrate in phase and fine spectral structure. This section provides the full diagnostic data behind that assessment.

#### 11.3.1 C.1 Waveform-Level Differences

The masked fraction is the portion of the difference energy falling below the auditory masking threshold, computed in the Bark domain with a −24 dB threshold.

The sample-by-sample waveform difference is large, and some test sentences have a negative SNR. But the frame energy envelope correlation stays between 0.956 and 0.969, indicating that the two have nearly identical prosodic structure and that the differences concentrate in phase and fine spectral structure. For short sentences, more than 99% of the difference energy falls below the auditory masking threshold.

#### 11.3.2 C.2 Accuracy of Automatic Quality Metrics

To determine whether the differences above are perceptually discriminable, different types of perturbation are applied to the same WORLD-synthesized audio and subjective perceptibility is compared against the change in automatic metrics.

Each row applies one type of perturbation to the same audio. Subjective perceptibility is judged by listening. The automatic metrics are the difference for the same audio before and after the perturbation.

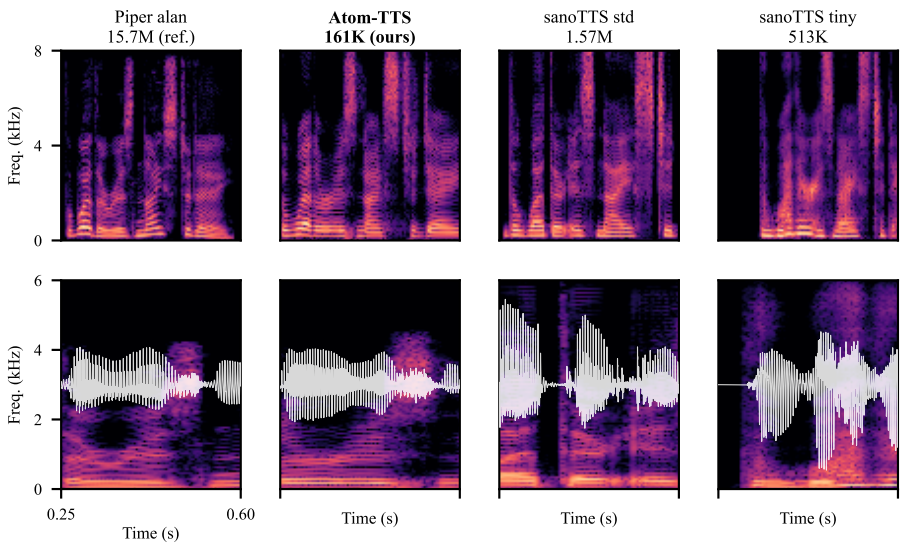


Figure A1: Spectrograms on the same sentence and time window: top, wideband (0–8 kHz); bottom, zoomed region (0.25–0.60 s, 0–6 kHz) with waveform; speaking rates differ, so the zoom window is taken from the start of each system's audio.

Table A8: Subjective perceptibility and automatic metric changes under different perturbations; perceptibility is judged by listening, and the metrics give the change for the same audio before and after the perturbation.

| Perturbation | Perceptibility | ΔSCOREQ | ΔUTMOS | ΔDNS |
|---|---|---|---|---|
| 8-bit quantization | Nearly inaudible | −0.154 | −0.030 | −0.009 |
| Added noise (30 dB) | Barely audible | −0.298 | −0.038 | −0.089 |
| **Actual FD-WORLD diff.** | **Inaudible** | −0.316 | −0.159 | −0.163 |
| Amplitude smoothing | Audible | −0.866 | −0.908 | −0.885 |
| Phase randomization | Nearly inaudible | −1.711 | −1.330 | −1.967 |

The three automatic metrics do not rank in the same order as subjective perceptibility. Phase randomization is almost imperceptible to listeners yet causes the largest metric drop, while the actual FD-WORLD difference corresponds to a metric drop smaller than that of perceptible amplitude spectrum smoothing. A rank correlation between subjective severity and metric change gives a Spearman coefficient of $-0.154$ ($p = 0.805$), meaning that none of the three metrics separates perceptible from imperceptible perturbations. The reason is that these metrics are sensitive to the phase dimension, which is exactly where FD-WORLD and WORLD differ.

#### 11.3.3 C.3 Within-Sentence Switching Experiment

Segments synthesized by WORLD and FD-WORLD within the same sentence are concatenated alternately at different intervals, and the response of the automatic metrics is observed.

The self-switching row serves as a control, confirming that the concatenation operation itself introduces no metric change.

The metric drop caused by 100 ms switching ($-1.221$) exceeds the difference of FD-WORLD itself from WORLD ($-0.568$), showing that this metric also penalizes the discontinuity introduced by concatenation. This further indicates that automatic metrics should not by themselves be used as a quality conclusion in comparisons of this kind.

Table A9: SCOREQ of within-sentence switching material

| Material | SCOREQ | Δ vs WORLD |
|---|---|---|
| WORLD unchanged | 3.518 | — |
| WORLD switched with WORLD (control) | 3.515 | −0.003 |
| FD-WORLD unchanged | 2.950 | −0.568 |
| 300 ms switching | 2.812 | −0.706 |
| 100 ms switching | 2.297 | −1.221 |

Table A10: Numerical consistency checks between the trained model and the deployed implementation

| Check item | Tolerance | Measured |
|---|---|---|
| Max absolute error per graph, PyTorch vs ONNX | $2 \times 10^{-4}$ | $\leq 8 \times 10^{-6}$ |
| Mean log-spectral difference, C++ vs Python reference | — | $< 0.3$ dB |
| Output of directory-asset mode vs single-file voice mode | — | bit-identical |

### 11.4 D. Implementation Consistency Verification

To ensure that the deployed implementation matches the trained model, three numerical comparisons were performed.

The FD-WORLD noise track is generated from a fixed seed. Because Python and C++ use different random number generators, sample-by-sample comparison is not applicable to the full synthesis chain, and verification instead uses energy, amplitude spectrum, and duration statistics.

**Cold-start latency and peak memory.** Table A11 reports the first-request latency and process peak working set of the two back ends under single-core conditions with intra-op threads set to 1. Each measurement uses a fresh process (including process startup, ONNX graph loading, and synthesis of the first sentence), repeated 5 times with the median taken and a fixed test sentence. FD-WORLD has a 33% lower cold start than WORLD and a slightly lower peak working set, consistent with its replacing the WORLD spectral envelope decomposition, a step that requires initializing a relatively large internal buffer. Together the two columns show that the benefit of FD-WORLD appears not only in steady-state throughput but also in first-request latency and memory footprint.

### 11.5 E. Items to Be Completed

## References

Rowel Atienza. EfficientSpeech: An on-device text to speech model. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2023. doi: 10.1109/ICASSP49357.2023.10094639.

Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017. doi: 10.1109/CVPR.2017.195.

Yann N. Dauphin, Angela Fan, Michael Auli, and David Grangier. Language modeling with gated convolutional networks. In *Proceedings of the 34th International Conference on Machine Learning*, 2017.

Hexgrad. Kokoro-82M: An open-weight text-to-speech model. `https://huggingface.co/hexgrad/Kokoro-82M`, 2024.

ITU-T. Recommendation p.800: Methods for subjective determination of transmission quality. International Telecommunication Union, 1996.

Table A11: Cold start and peak memory; single core with intra-op threads set to 1, each measurement using a fresh process that includes process startup, graph loading, and synthesis of the first sentence, with the median of 5 runs and a fixed first sentence.

| Back end | Median cold start | Cold-start range | Peak working set |
|---|---|---|---|
| WORLD | 179.4 ms | [178, 210] ms | 64.4 MB |
| **FD-WORLD** | **120.1 ms** | **[118, 150] ms** | **60.6 MB** |


Nal Kalchbrenner, Erich Elsen, Karen Simonyan, et al. Efficient neural audio synthesis. In *Proceedings of the 35th International Conference on Machine Learning*, 2018.

Takuhiro Kaneko, Kou Tanaka, Hirokazu Kameoka, and Shogo Seki. istftnet: Fast and lightweight mel-spectrogram vocoder incorporating inverse short-time fourier transform. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 6197–6201, 2022.

Jaehyeon Kim, Jungil Kong, and Juwoo Son. Conditional variational autoencoder with adversarial learning for end-to-end text-to-speech. In *Proceedings of the 38th International Conference on Machine Learning*, 2021.

KittenML. KittenTTS: A lightweight text-to-speech model. `https://github.com/KittenML/KittenTTS`, 2025.

Jungil Kong, Jaehyeon Kim, and Jaekyung Bae. Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

Feiyang Luo, Xu Tan, Yi Wang, Tao Qin, Sheng Li, Zhou Zhao, Enhong Chen, and Tie-Yan Liu. Lightspeech: Lightweight and fast text to speech with neural architecture search. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2021. doi: 10.1109/ICASSP39728.2021.9414403.

Michael McAuliffe, Michaela Socolof, Sarah Mihuc, Michael Wagner, and Morgan Sonderegger. Montreal forced aligner: Trainable text-speech alignment using kaldi. In *Proceedings of Interspeech*, pp. 498–502, 2017. doi: 10.21437/Interspeech.2017-1386.

Shivam Mehta, Rohan Tu, Jonas Beskow, Erika Szëkely, and Gustav Eje Henter. Matcha-tts: A fast tts architecture with conditional flow matching. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2024.

Masanori Morise. Cheaptrick, a spectral envelope estimator for high-quality speech synthesis. *Speech Communication*, 67:1–13, 2015. doi: 10.1016/j.specom.2014.09.003.

Masanori Morise. D4c, a band-aperiodicity estimator for high-quality speech synthesis. *Speech Communication*, 84:57–69, 2016. doi: 10.1016/j.specom.2016.09.001.

Masanori Morise. Harvest: A high-performance fundamental frequency estimator from speech signals. In *Proceedings of Interspeech*, pp. 2706–2710, 2017. doi: 10.21437/Interspeech.2017-68.

Masanori Morise, Fumiya Yokomori, and Kenji Ozawa. World: A vocoder-based high-quality speech synthesis system for real-time applications. *IEICE Transactions on Information and Systems*, E99-D(7):1877–1884, 2016. doi: 10.1587/transinf.2015EDP7457.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In *Proceedings of the 40th International Conference on Machine Learning*, pp. 28492–28518, 2023.

Chandan K. A. Reddy, Vishak Gopal, and Ross Cutler. DNSMOS P.835: A non-intrusive perceptual objective speech quality metric to evaluate noise suppressors. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 886–890, 2022. doi: 10.1109/ICASSP43922.2022.9746108.

Yi Ren, Yangjun Ruan, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. Fastspeech: Fast, robust and controllable text to speech. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Yi Ren, Chenjun Hu, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. Fastspeech 2: Fast and high-quality end-to-end text to speech. In *International Conference on Learning Representations*, 2021.

Rhasspy. Piper: A fast, local neural text to speech system. `https://github.com/rhasspy/piper`, 2023.

Takaaki Saeki, Detai Xin, Wataru Nakata, Tomoki Koriyama, Shinnosuke Takamichi, and Hiroshi Saruwatari. UTMOS: UTokyo-SaruLab system for VoiceMOS challenge 2022. In *Proceedings of Interspeech*, pp. 4521–4525, 2022. doi: 10.21437/Interspeech.2022-439.

Hubert Siuzdak. Vocos: Closing the gap between time-domain and fourier-based neural vocoders for high-quality audio synthesis. In *International Conference on Learning Representations*, 2024.

Ashish Thapa. sanoTTS: The smallest real-time neural TTS on a general-purpose microcontroller. arXiv:2608.21378, 2026.

TinyTTS Contributors. TinyTTS: Ultra-lightweight text-to-speech. `https://www.npmjs.com/package/tiny-tts`, 2025.

Keiichi Tokuda, Takao Kobayashi, Takashi Masuko, and Satoshi Imai. Mel-generalized cepstral analysis — a unified approach to speech spectral estimation. In *Proceedings of the International Conference on Spoken Language Processing*, pp. 1043–1046, 1994.

Keiichi Tokuda, Takashi Masuko, Noboru Yamada, Tetsunori Kobayashi, and Satoshi Imai. An algorithm for speech parameter generation from continuous mixture HMMs with dynamic features. In *Proceedings of Eurospeech*, pp. 1315–1318, 1995. doi: 10.21437/eurospeech.1995-173.

Jean-Marc Valin and Jan Skoglund. Lpcnet: Improving neural speech synthesis through linear prediction. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 5891–5895, 2019. doi: 10.1109/ICASSP.2019.8683756.

Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. In *arXiv preprint arXiv:1609.03499*, 2016.

Yuxuan Wang, R. J. Skerry-Ryan, Daisy Stanton, et al. Tacotron: Towards end-to-end speech synthesis. In *Proceedings of Interspeech*, pp. 4006–4010, 2017. doi: 10.21437/Interspeech.2017-1452.

Zhizheng Wu, Oliver Watts, and Simon King. Merlin: An open source neural network speech synthesis system. In *Proceedings of the 9th ISCA Speech Synthesis Workshop*, pp. 202–207, 2016. doi: 10.21437/SSW.2016-33.

Heiga Zen, Keiichi Tokuda, and Alan W. Black. Statistical parametric speech synthesis. *Speech Communication*, 51(11):1039–1064, 2009. doi: 10.1016/j.specom.2009.04.004.