Digital Signal Processing for Music

Part 16: Real-time and Blocking

Andrew Beck

Introduction

- Many audio processing systems are real-time systems
- This includes
  - Most audio plugins,
  - Studio Hardware effects, etc

Real-Time System (Wikipedia)

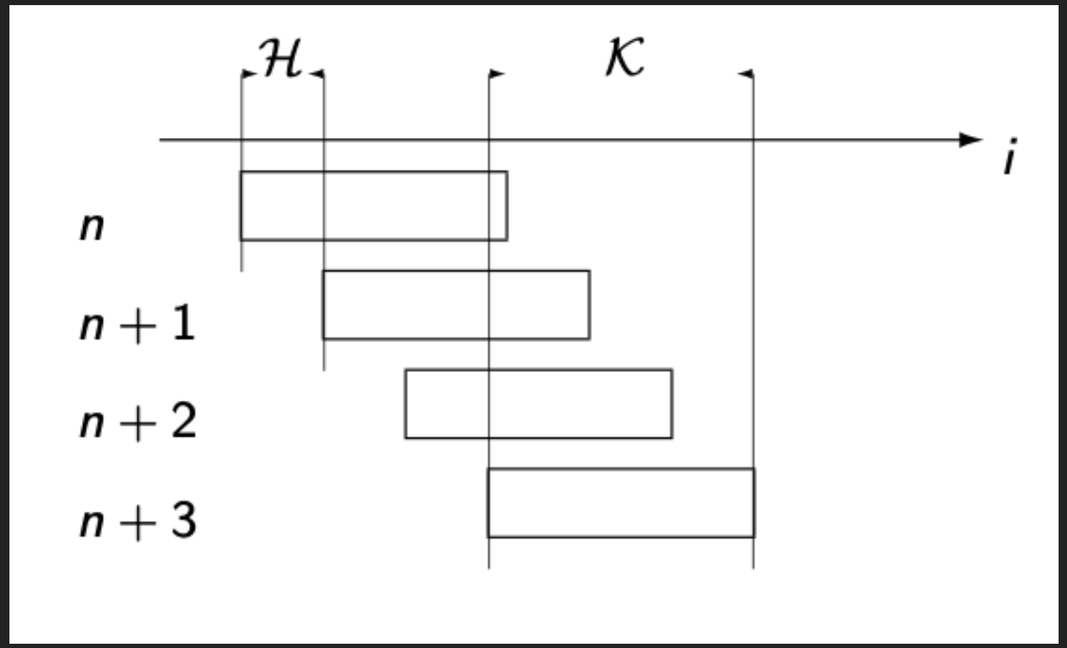
In a real-time digital signal processing (DSP) process, the analyzed (input) and generated (output) samples can be processed (or generated) continuously in the time it takes to input and output the same set of samples independent of the processing delay

- "processing delay and resources must be bounded even if the processing continues for an unlimited time"
- "mean processing time per sample is no greater than the sampling period, which is the reciprocal of the sampling rate"
- "perform all computations continuously at a fast enough rate that the output (...) keeps up with changes in the input signal"

Properties

Block based Processing

Processing of *blocks of samples* vs. individual samples



Reasons

- Block based algorithms (FFT, …)
- Audio hardware characteristics
- Efficiency (SIMD, memory allocation)

Block Sizes

- Typical block sizes can range from 1…thousands of samples
- Often powers of 2
- In many DAWs and some drivers the **block size varies**

Implications of Real-Time Systems on Effects

Can pitch shifting theoretically be implemented as real-time system?

Can time stretching theoretically be implemented as real-time system?

In-Place Processing

Samples of the input block are replaced with output block

Pro

- Resource friendly, memory allocation for output buffer

Con

- Original input data connot be used anymore

Blocking

- **Time-Stamps**
  - Blocking can be considered similar to down-sampling
  - *What time stamps to assign to each block?*
    - Beginning of each block
    - Center of each block
- **Initialization**
  - Real-time systems are designed to work for infinite input stream
  - *How to initialize internal buffers?*
    - Usually zeros, but other initializations may make sense in specific scenarios
- **Performance** issues due to blocking
  - Plugin gets stream of samples split into small blocks (e.g., 32 samples)
  - Internally, STFT with large hopsize (e.g., 2048 samples) is used
  - *What is the potential performance problem here?*
    - Each hop requires data from 64 input blocks
    - No processing can be done for 63 blocks
    - Processing of huge FFT has tro be done during the 64th block (32 samples)

Maintaining Constant Time Processing

- Calling functions with inconsistent execution time
  - Particularly *malloc* and *free*
  - Using locks to communicate across threads

Summary

Real-Time systems have the following properties

- Hard **performance** requirements
  - Processing of input block has to be faster than time span of this block **for all blocks, not only on average**
- **Causality**
  - Future samples cannot be taken into account (or only by increasing the latency: Look-ahead)
- **Latency**
  - Time between input and system response, usually intended to be minimal